

Thesis no: MSCS-2014-04



Implementation and evaluation of global router for Information-Centric Networking

A real time implementation

Yogaraj Baskaravel

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona, Sweden

This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Computer Science. The thesis is equivalent to 20 weeks of full-time studies.

Contact Information:

Author:

Yogaraj Baskaravel

E-mail: yoba10@student.bth.se

External advisor:

Dr. Bengt Ahlgren

SICS Swedish ICT AB

University advisor:

Prof. Adrian Popescu

Dept. Communication Systems

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona, Sweden

Internet : www.bth.se
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

Abstract

Context. A huge majority of the current Internet traffic is information dissemination. Information-Centric Networking (ICN) is a future networking paradigm that focuses on global level information dissemination. In ICN, the communication is defined in terms of requesting and providing Named Data Objects (NDO). NetInf is a future networking architecture based on Information-Centric Networking principles.

Objectives. In this thesis, a global routing solution for ICN has been implemented. The authority part of NDO's name is mapped to a set of routing hints each with a priority value. Multiple NDOs can share the same authority part and thus the first level aggregation is provided. The routing hints are used to forward a request for a NDO towards a suitable copy of the NDO. The second level aggregation is achieved by aggregating high priority routing hints on low priority routing hints. The performance and scalability of the routing implementation are evaluated with respect to global ICN requirements. Furthermore, some of the notable challenges in implementing global ICN routing are identified.

Methods. The NetInf global routing solution is implemented by extending NEC's NetInf Router Platform (NNRP). A NetInf testbed is built over the Internet using the extended NNRP implementation. Performance measurements have been taken from the NetInf testbed. The performance measurements have been discussed in detail in terms of routing scalability.

Results. The performance measurements show that hop-by-hop transport has significant impact on the overall request forwarding. A notable amount of time is taken for extracting and inserting binary objects such as routing hints at each router.

Conclusions. A more suitable hop-by-hop transport mechanism can be evaluated and used with respect to global ICN requirements. The NetInf message structure can be redefined so that binary objects such as routing hints can be transmitted more efficiently. Apart from that, the performance of the global routing implementation appears to be reasonable. As the NetInf global routing solution provides two levels of aggregation, it can be scalable as well.

Keywords: Information-Centric Networking, routing, future networking.

Acknowledgements

This thesis was carried out at the center for networked systems at Swedish Institute of Computer Science (SICS) and as part of the EU-funded research project Scalable and Adaptive Internet Solutions (SAIL). Foremost, I would like to express my profound gratitude to my advisor Bengt Ahlgren for giving me this opportunity. His guidance throughout the time of research and constant support while writing this report have been enormous. I would also like to thank Prof. Adrian Popescu who is my advisor at Blekinge institute of technology for patiently reviewing the report.

I want to acknowledge Anders Lindgren and the fellow researchers at the center for networked systems for the stimulating discussions during my thesis. I am grateful to Dirk Kutcher of NEC for providing the NEC's NetInf Router Platform that is used in the thesis implementation. Special thanks to Börje Ohlman and Karl-åke persson of Ericsson and Bruno Kauffinen of France Telecom for their support in setting up the experiment testbed.

I would like to thank my parents Baskaravel and Thilagavathi and my brothers Balaji and Sakthivel for their love and encouragement especially during my master studies. Last but not the least, I am thankful to all my friends for their support in all my endeavors.

Yogaraj Baskaravel

Contents

Abstract	i
Acknowledgements	ii
Contents	iii
List of Figures	v
List of Tables	vi
List of Acronyms	vii
1 Introduction	1
1.1 Problem Definition	2
1.2 Aims and Objectives	3
1.3 Research Questions	4
1.4 Expected Outcomes	4
1.5 Research Methodology	4
1.6 Thesis Outline	5
2 Background	6
2.1 Host-centric Networking	6
2.2 Information-centric Networking	7
2.3 NetInf	7
2.4 Other Approaches	9
3 NetInf Global Routing	10
3.1 Routing Scalability	10
3.2 NetInf Routing Challenges	12
3.2.1 Naming	12
3.2.2 Routing	14
3.3 NetInf Global Routing	16
3.3.1 Naming Format	16
3.3.2 Hint Lookup	17
3.3.3 Routing Hints	18

3.3.4	Forward Lookup	19
3.4	Forwarding Steps	20
4	Implementation	21
4.1	Build Methodology	21
4.2	NEC NetInf Router Platform	21
4.2.1	Message Structure	22
4.2.2	Basic Concepts	22
4.3	NetInf Global Routing	24
4.3.1	Processing Sequence	24
4.3.2	HTTP Convergence Layer Module	26
4.3.3	Hint Lookup Module	26
4.3.4	Forward Lookup Module	28
4.4	Licensing	30
4.5	Observation	30
5	Experiment	31
5.1	Experiment Design	31
5.1.1	Experiment Setup	31
5.1.2	Measured Node	32
5.1.3	Pilot Study	33
5.1.4	Validity Threats	33
5.2	Performance Evaluation	34
5.2.1	Statistical Significance	34
5.2.2	Overall Forwarding Time	34
5.2.3	Forwarding Steps	36
5.2.4	Extracting Hints	37
5.2.5	Hint Lookup	38
5.2.6	Merging Hints	38
5.2.7	Forward Lookup	40
5.2.8	Frame Forward	40
5.2.9	DNS Hint Lookup	41
5.3	Observation	43
6	Conclusion and Future work	44
6.1	Conclusion	44
6.2	Future Work	46
	Bibliography	47

List of Figures

2.1	Name Based Routing in NetInf	8
2.2	Name Resolution Service in NetInf	9
3.1	sample NDO name	16
3.2	Routing hint lookup	17
3.3	Hint lookup in edge network and DFZ	17
3.4	Sample ext field with routing hints	18
4.1	NNRP message Structure	22
4.2	NNRP chains and modules	23
4.3	NNRP NetInf global routing	25
4.4	Hint lookup processing flow	27
4.5	Forward lookup processing flow	29
5.1	Experiment NetInf Testbed	32
5.2	Frequency distribution of overall forwarding time	35
5.3	Overall forwarding time vs nexthop connection time	35
5.4	Median time taken by GET request forwarding steps	37
5.5	Time taken to extract hints	38
5.6	Time taken to perform hint lookup	39
5.7	Time taken to merge new routing hints to existing routing hints	40
5.8	Time taken to perform forward lookup	41
5.9	Time taken to frame and forward GET request	42
5.10	Time taken to perform dns hint lookup	42

List of Tables

3.1	Locator forwarding table	19
5.1	Nee-wom system information	33
5.2	NetInf global routing steps	36

List of Acronyms

- ASCII** American Standard Code for Information Interchange.
- BGP** Border Gateway Protocol.
- CCN** Content-Centric Networking.
- CDN** Content Delivery Network.
- CIDR** Classless Inter-Domain Routing.
- COMET** Content mediator architecture for content aware networks.
- DFZ** Default Free Zone.
- DNS** Domain Name System.
- DONA** Data Oriented Network Architecture.
- DPI** Deep Packet Inspection.
- HTTP** Hypertext Transfer Protocol.
- IAB** Internet Architecture Board.
- ICN** Information-Centric Networking.
- IP** Internet Protocol.
- IPv4** Internet Protocol version 4.
- IPv6** Internet Protocol version 6.
- ISP** Internet Service Provider.
- JSON** JavaScript Object Notation.
- LISP** Locator Identifier Split.

NBR Name Based Routing.

NDN Named Data Networking.

NDO Named Data Object.

NetInf Network of Information.

ni named information.

NNRP NEC NetInf Router Platform.

NRS Name Resolution Service.

P2P Peer to Peer.

PA Provider Allocated.

PI Provider Independent.

POSIX Portable Operating System Interface.

PSIRP Publish-Subscribe Internet Routing Paradigm.

PURSUIT Publish-Subscribe Internet Technology.

RIB Routing Information Base.

RRG Routing Research Group.

SAIL Scalable and Adaptive Internet Solutions.

SICS Swedish Institute of Computer Science.

TCP Transmission Control Protocol.

URI Uniform Resource Identifier.

Chapter 1

Introduction

As the number of users and devices connected to the Internet is keep on increasing, many new kind of internet applications are being introduced in order to facilitate content distribution. The amount of content shared among the internet users is ever increasing, finding efficient and scalable content distribution mechanisms is an interesting research problem.

The main abstraction of the Internet and mobile systems is named hosts, and communication is defined in terms of end-to-end data transmission between these hosts. This design can be defined as Host-Centric Networking. Earlier, when current network architectures were developed, focus was on connecting one host to another and share expensive and limited resources [1]. However the evolution of digital information has highly influenced the data communication over Internet. This includes retrieving audio/video files, multimedia streaming, browsing web pages and many more. Jacobson et al [1]. pointed out that the huge majority of current Internet traffic is information dissemination rather than end-to-end communication. Since Host-centric Networking is not designed to serve information dissemination, Internet and mobile systems face various architectural issues.

To overcome the architectural issues in the present Internet, Jacobson and others [1, 2] argued that future Internet should be designed focusing on named data rather than named hosts. They therefore propose Information-Centric Networking (ICN) built on named data as the main abstraction. Named Data Object (NDO) refers to actual data objects transmitted over the network. Communication is defined in terms of requesting and publishing NDOs with integrated caches in the network infrastructure. An NDO can refer to audio, video, web page, image, email and many other data objects, each mapped to unique name identifiers.

NetInf is a future network architecture based on the ICN principles. It was initially developed in the Architecture and Design for the Future Internet (4WARD) project [3]. It was then carried forward in the SAIL project [4]. Naming Data Objects, securing NDOs, caching NDOs in network and name resolution/routing are the four main components of NetInf.

NDO retrieval in ICN can be done with two different methods. In the first method, which is known as Name Resolution Service (NRS), the NDO name is resolved into NDO's network location. Once the network location is retrieved, a request for the NDO is sent to the identified network location and response with NDO is sent back to requester using legacy routing protocols. In the second method, which is known as Name Based Routing, NDO requests are routed to NDOs' locations based on NDO names in the network. Each router contains next hop mapping for each name [2]. Since traditional routing mechanisms are not sufficient to handle requirements of future Internet, routing mechanisms that combine name resolution service and name based routing are being investigated [2].

Building an NRS system for a global ICN network is challenging due to enormous amount of NDOs. Though there are some proposals to provide NRS on a global level, NDO requests are still forwarded using legacy routing protocols. Hence it will be difficult to apply ICN functions on forwarded path. For instance, NDO requests are routed in a cache unaware manner i.e., router caches are not looked up for the requested NDO. This brings us to Name Based Routing that performs cache aware routing i.e., NDO is served from a router if it is available in the router's cache. Due to the same reason as NRS, building global level Name Based Routing is challenging. However there are a few solutions that divide routing problem with respect to three network domains i.e., ingress edge, core and egress edge [2]. In other words, the routing from a client to a service provider network, within the service provider networks and from a service provider network to a server. Considering size of core network and requirements to handle huge amount of NDO requests, it is worth to look into scalable routing solution for ICN core networks.

1.1 Problem Definition

Routing is important in all kinds of networks and especially scalability of routing needs special attention [5, 6]. Name based routing in ICN makes it harder compared to Internet Protocol (IP) routing, as it is required to maintain enormous amount of routing entries. Data objects can be named using hierarchical names or flat names. Flat names have been highly criticized for their scalability and security [7]. Aggregation of routing information is

inevitable with either case of naming schemes in order to build a scalable routing system. There are some proposals that make use of aggregation in Name Based Routing that uses flat names. Ghodsi et al. [7] claim that flat names have better security and scalability properties than hierarchical, human-readable names. Explicit aggregation can be performed on any given set of globally unique names by concatenating the names. Routing is performed using each of the concatenated names [7]. Notably Narayanan and Oran [8] pointed out that there is no way to scale up Border Gateway Protocol (BGP) in the near future to satisfy increased scaling requirements of present and future communications. In addition, they state that BGP cannot scale up to the expected size of name based routing table even with various aggregation methods. They proposed a solution that translates NDOs' names into routing labels that do not refer to any host. Routing is performed using the routing labels retaining the topological properties [8].

Inspiring from the above solution, NetInf Global Routing using routing hints has been described focusing on global Information-Centric Network [9]. Similar to routing labels, NetInf Global Routing introduces routing hints that use the Internet Protocol version 4 (IPv4) name space, but treats it as a flat name space without the notion of network prefix. This variant of NetInf router performs cache aware routing of NDO requests. Scalability requirement of name based routing especially in core network domain needs to be solved with an efficient routing mechanism. NetInf global routing is argued to be scalable with respect to core network domain [9].

1.2 Aims and Objectives

This research work aims to implement and evaluate NetInf global routing and to perform a performance and scalability analysis of the implementation. The performance and scalability analysis is performed in terms of the time taken by the various steps in forwarding a GET request. The following are the objectives of this thesis.

- NEC NetInf Router Platform (NNRP) shall be extended to provide mapping from NDO's domain name into set of routing hints. The mapping shall be configured statically and can also be retrieved from Domain Name Server.
- Static routing can be implemented to route using routing hints.
- Forwarding mechanism shall be implemented to forward NDO requests based in routing hint to next hop address mapping.
- Performance and scalability of the implementation are to be evaluated taking global ICN routing requirements into account.

1.3 Research Questions

The following are the research questions that drive this research.

- What are the challenges in implementing NetInf global routing?
- What are the performance and scaling properties of NetInf global routing?

1.4 Expected Outcomes

This research work aims to implement and observe a global routing scheme for ICN. Furthermore, challenges in implementing ICN routing shall be identified. In addition, providing feedback on NetInf protocol specifications, identifying additional protocol requirements and interoperability issues with other NetInf functionalities can be carried out. NNRP [9] will be extended with NetInf global routing. SAIL NetInf testbed will be built which connects NetInf nodes in SAIL partner sites across Europe. Extended NNRP implementation shall be experimentally observed in the SAIL NetInf testbed and scalability metrics will be evaluated with respect to global ICN network.

1.5 Research Methodology

Research methodology [10] is a procedure of carrying out a research work in a systematic way. The first research question relates to the implementation of the NetInf global routing solution. The NetInf global routing solution will be implemented by extending the NNRP software. The challenges in implementing NetInf global routing can be identified while extending the NNRP software. Build research methodology is suitable for building a new software system or extending an existing software system. Hence, build research methodology had been chosen to answer the first research question.

The NetInf testbed will be built on top of the Internet using the extended NNRP software. Experimental methodology is selected to answer the second research question. Experimental methodology consists of an exploratory phase and an evaluation phase. The time taken by various steps in the overall GET request forwarding are measured during the exploratory phase. The evaluation phase aims to identify the performance and scaling properties of NetInf global routing from the measurements.

1.6 Thesis Outline

Chapter 1 presents the research problem and formulates research questions that drive this thesis work. Background of Host-centric networking, Information-centric Networking and NetInf are explained in chapter 2. Also, it lists problems in host-centric networking, ICN solutions and building blocks of NetInf. A systematic literature review of routing scalability in current networks and ICN is presented in chapter 3. It also discusses NetInf Global Routing in detail. NNRP overview, design of extended implementation, Domain Name System (DNS) support for routing hints and finally licensing details are covered in chapter 4. Experiment and performance evaluation are presented in chapter 5. This includes scalability analysis of the measurements. Chapter 6 presents the conclusion and future works.

This chapter briefly presents the evolution of the current Internet from telephone networks. Also, it describes core concepts of Information-Centric Networking (ICN) and how it can overcome some of the architectural issues in the current Internet. Finally it introduces Network of Information (NetInf), a network architecture based on ICN principles.

2.1 Host-centric Networking

Traditional telephone networks were designed to transfer voice signals between two phones. The telephone networks used end-to-end electric circuits to exchange voice signals. Once connection is established between two phones, voice signals are exchanged over the connection. The telephone networks were not suitable for data communication due to the limitations in setting up the connection path. Hence, the focus was moved from connection paths to connection end points in order to support data communication.

In earlier days of developing digital networks, the switching elements were defined to store and forward standard packets based on destination host information [11]. Named hosts were defined as the main abstraction of this architecture and communication is defined in terms of information exchange between two hosts. The current Internet architecture is using the same basic model irrespective of the growing demands of the Internet. Information dissemination constitutes a major portion of the current Internet traffic. Enormous amount of applications are being used and developed for mass data consumption.

The increased amount of data objects and the distributed retrieval of data objects pose huge challenges on content providers and network infrastructure. These challenges are being addressed by application layer overlays such as CDN and P2P applications [12]. However these applications are still host centric and adds complexity to the communication. The connection between a requester and data objects are decided and maintained by end hosts instead of the

network itself. For example, retrieval of a NDO from a nearby CDN node is decided by an application end host instead of a router. Some sort of network intelligence such as Deep Packet Inspection (DPI) is required just to identify the type of data being transmitted. Since DPI requires significant computation and breaks fundamental host centric communication, DPI can not be performed at each router. Hence, routing based on data objects is not feasible. In addition, low layer protocols can not apply their feature sets in an effective way. For example, wireless networks can not apply broadcast and multicast features for disseminating same data objects to multiple receivers. Since data objects are location dependent, multihoming or mobility of nodes increases the complexity in application overlays and domain name servers.

2.2 Information-centric Networking

ICN [2] is a future networking paradigm that is being developed under various research projects. ICN aims to address the challenges faced by current networking architecture and also address future networking requirements. One of the main focus of ICN is to distribute Named Data Objects (NDOs) on a global scale by having NDOs as the main abstraction. The main components of ICN include naming, security, routing and caching.

Location-independent names are used to identify NDOs, that is, the name remains irrespective of the NDO's location. Security of communication is tied to the NDO itself, enabling secure retrieval from an untrusted location. Routing component enables the communication between requesting host and NDO's network location. Routers forward a NDO request from a requesting host towards a suitable copy of the requested NDO in the network. ICN enables integration of caches for NDOs as part of the normal network service. Hence consecutive requests to the same object can be served from network caches itself.

2.3 NetInf

NetInf is a future networking architecture based on ICN principles, that aims to provide global scale communication [12]. NetInf was used in this research work, as it implements a routing solution for global ICN. NetInf nodes enable communication to access NDOs using their names. A flat name space is used for naming NDOs in order to avoid complexities in moving data objects in a hierarchical namespace. NDO names are used in routing as well as for validating NDO's integrity.

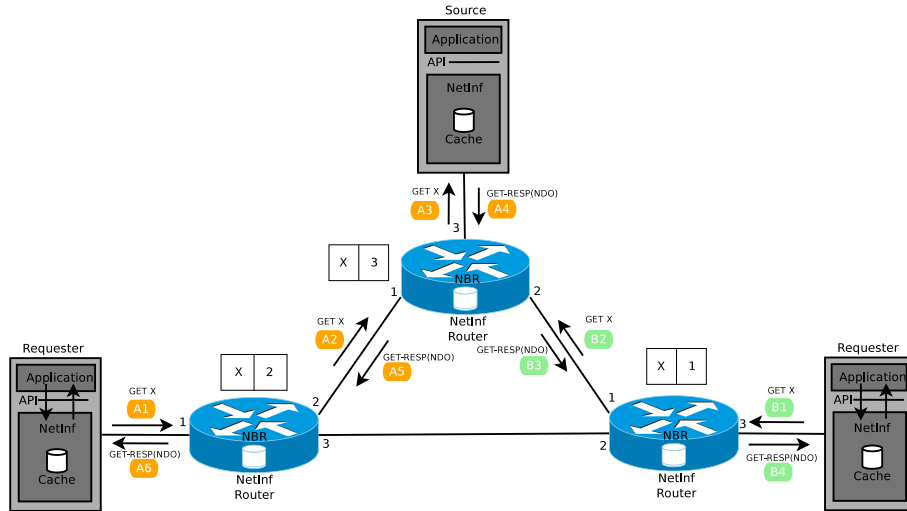


Figure 2.1: Name Based Routing in NetInf

Name Based Routing (NBR) and Name Resolution Service (NRS) are two ways of forwarding NDO requests towards NDO's location. In NBR, each NetInf node contains name to nexthop mapping. This mapping is used to forward NDO requests towards its location. In NRS, NDO requestor first queries name resolution server and gets corresponding NDO's location. The NDO request is then sent to the identified NDO location.

NetInf supports on-path and off-path caching in order to reduce the latency of consecutive requests to same NDOs. In addition, it supports various convergence layer approaches to run NetInf on top of current transport protocols. It also provides flexibility to perform custom configurations for heterogeneous network requirements. Since NetInf networks are aware of contents that are being requested, a node can route NDO requests to the nearest source that has the requested NDO.

The NetInf protocol defines fundamental message types for NetInf Communication. A request for an NDO is defined as a GET message and the response to this request is defined as a GET-RESP message. Each GET message is associated with a message id which is returned by the corresponding GET-RESP message.. There are also message types for publishing NDOs into networks and for searching NDOs using keywords. Readers are referred to NetInf protocol specification [13] for more information on these message types. Using NBR, GET messages are routed towards a NDO's source location based on the NDO's name. Figure 2.1 shows NBR of a GET request in which X denotes the requested NDO's name. A GET-RESP message containing NDO X

is sent back to the requester, during which routers on the path can optionally cache the NDO. Request sequences A and B denote consecutive GET requests to the same NDO from two different requesters. Since the NDO is cached in one of the routers during request sequence A, the GET request in sequence B for the same NDO is served from the router cache. GET-RESP message routing should be done in a flexible way so that connection states in routers that forwarded the GET request are handled properly. The GET-RESP message routing can simply be achieved by using same socket in which GET message was received to send back GET-RESP.

In Name resolution service, a GET request is first sent to a Name resolution server, which responds back with GET-RESP containing network locators. The requester then sends a new GET request to the obtained location and receives a GET-RESP message with the requested NDO. NDO retrieval using name resolution service is shown in Figure 2.2.

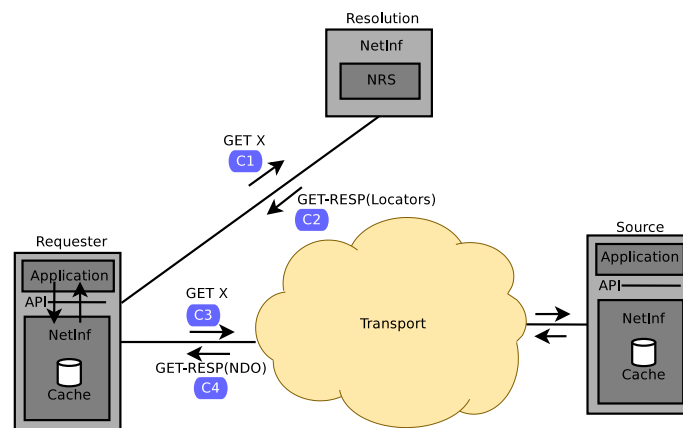


Figure 2.2: Name Resolution Service in NetInf

2.4 Other Approaches

Numerous other research projects are involved in the development of the future Internet architecture based on ICN principles. Named Data Networking (NDN) which was previously known as Content-Centric Networking (CCN) [1], Publish-Subscribe Internet Technology (PURSUIT) project that took over Publish-Subscribe Internet Routing Paradigm (PSIRP) [14], Data Oriented Network Architecture (DONA) [15] and Content mediator architecture for content aware networks (COMET) [16] are some of the notable ICN projects. Though this research work uses NetInf approach, the design is applicable to other ICN approaches as well.

Chapter 3

NetInf Global Routing

This chapter presents the challenges in scalable routing of host-centric networks and information-centric networks. First, it explains the related research works on scalability issues in global Internet routing. After that, the challenges in scaling name-based routing and name resolution service are presented. Finally, it presents NetInf global routing solution in detail.

3.1 Routing Scalability

The routing scalability is the ability to scale the routing system to cope with the increased number of destination nodes. The basic functionality of a router in a host-centric network is to route packets based on its destination address. The routers store a significant number of locator to next-hop mapping in the so called routing table that will be looked up to route a packet. The routing system aggregates destination hosts on their addresses prefix i.e. routing entries are stored only for prefixes rather individual destination addresses.

Internet routing has been facing scalability challenges right from earlier days of the Internet. Solutions such as Classless Inter-Domain Routing (CIDR) have been used to address scalability problems to a certain extent. The report from the IAB workshop [17], discussed pressing issues on the Internet routing and addressing system. Furthermore, it set a plan on developing scalable routing and addressing system. The workshop report focuses mainly on the issues in the Default Free Zone (DFZ). This refers to the network of core routers in the Internet that contain complete routing information. This routing information is sufficient to route packets to any global destination.

The present size of Routing Information Base (RIB) in DFZ is approximately 4.9×10^5 routes which is significantly large and keeps on growing. Wide applications of multihoming, traffic engineering and non-aggregatable networks are the main reasons for rapid growth of DFZ RIB [18]. These factors significantly reduce the amount of route aggregation on address prefixes and in turn increase the amount of non-aggregatable prefixes in the core networks. As

a result of increasing non-aggregatable routes in DFZ, the dynamics of edge networks directly affects the core networks. In other words, an unstable edge network will result in a lot of BGP updates in the core network. Though the number of routes is restricted by the address space size, the routing protocol dynamics such as amount of BGP churn messages in the core networks are not limited.

Most of the larger enterprises started using Provider Independent (PI) addresses rather than Provider Allocated (PA) addresses. PI addresses give enterprises the flexibility in changing their Internet Service Provider (ISP). PI addresses avoid network renumbering and in turn access control list modifications. However, PI addresses produce a lot of non-aggregatable routing entries in the core networks. Multihoming is another reason for the increase in DFZ RIB. Multihoming enables sites to be served through multiple ISPs and provides connection redundancy. Multihoming works using PA addresses as well as PI addresses. However, both of the approaches result in increased routing entries in the core networks.

Since there are various services provided over the Internet, the traffic in networks is increasingly complex. ISPs perform traffic engineering in order to achieve load balancing, cost reduction and policy enforcement. For example, an edge network that is connected through multiple service providers can load balance the upstream traffic among the service providers. This results in injecting more routing prefixes to the core network which becomes a routing scalability issue [19]. Moreover, edge networks can split routing prefixes in order to distribute the traffic.

In the current Internet, IP address space is overloaded to play two different roles. These roles are commonly known as locators and identifiers. There are some proposals such as Locator Identifier Split (LISP) that separates locating a host and identifying a host. Such solutions shall provide scalable routing mechanisms independent of host identification. As the global Internet is growing rapidly, IPv6 address space provides an increased address space and in turn increases DFZ RIB. There are various solutions to deploy IPv6 along with IPv4, almost all of them results increased RIB.

Workshop report presents the priority list of problems on the Internet [17]. Routing scalability takes the highest priority, as mobile technologies enable rapid increase in the amount of Internet users. Also, concepts like Internet of things will accelerate the increase in number of nodes connected to the Internet. Inter-domain scalability, mobility, multihoming and inter-domain traffic engineering scalability are the important challenges faced by the present Internet routing and addressing system. Routing Research Group (RRG) is

intended to create an alternate architecture in order to address these challenges [20]. The group has presented a prioritized list of design goals that can be used as rough guidelines to develop new routing and addressing system. As expected, the highest priority design goal is to provide routing scalability. As the global BGP routing table has kept on growing over the years [21], various research works are repeatedly stressing on the need to provide routing scalability. It is highly desired to have a routing system that can scale independent of user population growth. Despite the differences between ICN routing and present Internet routing, the above stated challenges need to be addressed in ICN as well.

3.2 NetInf Routing Challenges

The present Internet uses IP addresses to route data packets. Similarly, NetInf can perform routing using NDO names. NDN's hierarchically structured names are good enough to be used by present routing protocols [22]. However, the number of NDOs in the present global network is enormously larger than the number of nodes in the present Internet. As discussed in the previous section, the present routing and addressing system is already facing many challenges in terms of scalability. Hence, using the existing routing mechanisms that can scale up to a global ICN network is really challenging.

3.2.1 Naming

Naming NDOs is a crucial function in ICN, since it optionally supports name data integrity, human readability and routing aggregation. Name data integrity is a way of verifying NDO's integrity from the NDO's name. Names are used for routing GET requests as well as validating NDOs. There are three types of naming schemes commonly used in ICN, they are hierarchical naming, flat naming and attribute value pair naming [23]. In case of ICN approaches such as DONA and PURSUIT, NDO's name include the cryptographic hash of the publisher's public key. On the other hand, NetInf includes the cryptographic hash of the NDO itself. Though it makes names unreadable for humans, it enables name data integrity and global uniqueness. In other words, NDO's integrity can be verified by comparing the hash from the NDO's name and the hash of the NDO.

Hierarchical human readable names support name-based aggregation and user friendliness. However it is difficult to provide global uniqueness and name data integrity. Though flat names can provide global uniqueness and name data integrity, their location-independent nature makes it difficult to retrieve nearby copy of a NDO [1]. Moreover, since they are not human readable, users may end up retrieving malicious NDO. Hence, Jacobson et al. [1] claim that there needs to be a mechanism to verify that users are getting the right NDO name.

Ghodsi et al. [7] responded to this criticism on flat names and argued that in fact better scalability and security can be achieved using flat names. A flat name can include the hash of the public key of the publisher or signing authority. Authors claim that this way of binding between flat name and publisher is much better than the binding between hierarchical human readable name and publisher. Though aggregation does not come with the structure of flat names itself, explicit aggregation can be constructed by concatenating the flat names.

For example A.B.C is the explicit aggregation constructed from globally unique names A, B and C. Routing entries of B can be found by means of routing to A and similarly routing entries of C can be found by routing to B. Explicit aggregation using flat names are more flexible than strict hierarchical aggregation. Aggregation using hierarchical names are strictly limited to sub-parts of itself. Whereas, explicit aggregation can be performed on various naming concatenations i.e., A.B.C and E.D.C. Deepest match of name concatenation is used for routing and the same name can be part of multiple aggregates. Names of third party content providers can be concatenated to a NDO name. Hence, GET requests can be routed to third party content providers prior to actual publisher. [7]

NetInf uses flat names and contains cryptographic hash in order to support name data integrity. NetInf names NDOs using named information (ni) Uniform Resource Identifier (URI) scheme that is defined in RFC 6920 [24]. A ni URI includes the cryptographic hash of the NDO and also the name of the hashing algorithm that was used to generate the hash. It also includes an authority part that refers to the NDO's publisher. NetInf mostly uses domain name to denote a publisher.

3.2.2 Routing

Scalable routing is one of the important research topics in Information-Centric Networking. In ICN, Routing should support unbounded namespace along with bounded routing states. Routing should also enable intelligent forwarding of GET requests over multiple paths [22]. Though ICN routing is still in the research phase, as the initial approach routing problem is divided into two types. They are intra domain routing and Inter-domain routing.

Since this paper presents global routing in ICN networks, the focus is on inter-domain routing. Notable research is ongoing in achieving name aggregation using provider assigned tree-like names akin to provider assigned IP address. Jacobson et al. states that these names can be mapped to underlying hyperbolic name space. It is theoretically supported that optimal routing performance can be achieved when using greedy routing.

Rajahalme et al [25] proposed a hybrid network design for name-based inter-domain routing. NDO owners and users are connected to local nodes that maintain NDO reachability states. A hierarchical interconnection overlay bridges these local nodes in order to form the global ICN network. Furthermore, local nodes are connected to the adjacent nodes. Thus a hybrid network architecture is formed. This network design makes use of a flat namespace to name scopes. A scope refers to an object collection that share common distribution and access semantics. The availability of scopes are advertised to adjacent nodes in the same level of hierarchy and top-level node for achieving global availability. Identifier prefixing is done to aggregate object collections within a domain. Request routing in this architecture is done in multiple phases. A requested NDO is searched in the local domain, adjacent domains and interconnection overlays in the same order. The global scalability is achieved by aggregating objects into object collections and aggregation on scope identifier prefix. The prefix is used for routing aggregation in interconnection overlays.

Narayanan and Oran [8] listed out the routing scalability challenges in IP networks and ICN. They also proposed an ICN routing solution for global ICN network that can scale better. Some of the ICN approaches use current routing protocols to perform content routing. As discussed in the previous chapter, routing scalability is the most important problem in the present Internet. Authors state that the scalability requirements on BGP have been significantly increased by MPLS VPN. Authors argue that it is hard to scale BGP to meet ICN requirements in the near future. ICN based on flat namespace would have to support $O(10^{12})$ routes which is hugely larger than BGP's magnitude. Hence the ICN routing table needs to be compressed. NDN aggregates name routes

using longest prefix match forwarding. However, it adds limitation to NDO names that name should start with unambiguous prefix. Names are also desired to be location independent. Hence, NDN names can not really depend on topological prefix binding. A better solution is needed to compress ICN routing table.

Domain Name System (DNS) names in the present Internet are global, unambiguous and location independent and there are efforts from NDN to use DNS names as NDO name prefixes. There are already 13×10^7 names assigned in generic top-level domains (gTLD) (.com, .net, .org, .net). Adding up country code top-level domain names (ccTLD) will result in total names of 2×10^8 . It is obvious that adding second level domain names will hugely increase this total. An assumption on scale free growth of contents in sub-level domain names shows that the total number domain name prefixes may reach 6×10^8 which is 200 times larger than current BGP [8].

The ICN routing table needs to be compressed at least in six orders of magnitude. Achieving the routing table compression on flat namespace is nice. Topological aggregation of location independent NDO names is hard. However topological aggregation is preferred by routing algorithms. ICN routing scalability is hard because it requires enormously huge amount of routers to store big sized routing table. Narayanan and Oran [8] proposed a routing solution that retain topological properties in routing on location independent NDO names.

NDO names are translated into a set of routing labels by performing translation lookup. Routing is performed using these labels and topological properties are retained. Hence, routing table can be compressed topologically but these labels do not denote any hosts. Routing does not follow any implicit hierarchy and thus routing is done using exact match on labels rather than longest prefix match. A NDO name can be translated to a partially ordered list of labels and routing is done using the label with highest precedence. Explicit aggregation is performed by aggregating low precedence labels on high precedence labels. In other words, low precedence labels are advertised in broader network whereas, high precedence labels are advertised only closer to the destination. [8]

3.3 NetInf Global Routing

In this section, the implemented NetInf global routing solution is explained. A scalable routing solution has been defined in the Global Information Network Architecture (GIN) [26]. This routing solution has global name resolution system that maps part of NDO names to locators. The locators are network domain identifiers that are used in BGP-style routing system similar to today's IP prefixes. Each GET request carry a label stack (*locator*) that implements explicit aggregation of routing information.

Inspiring from Narayanan and Oran's solution [8], a new variant of GIN global routing is described in SAIL project deliverable. NetInf global routing solution is developed based on the routing variant. Similar to the present Internet, NetInf networks can be divided as core network and edge network. NetInf global solution is mainly focused on the routing in NetInf core network. In other words, routing of GET requests in NetInf DFZ similar to DFZ in present Internet [9].

The thesis work implements and evaluates NetInf global routing solution. Hereafter, NetInf global routing solution is referred as the global routing solution. NetInf Global Routing is a name-based routing mechanism that introduces Routing Hints. In NetInf global routing, names are translated into routing hints. The routing hints are used in routing NDO requests towards NDO's location. In this section, ni naming format is briefly described. Furthermore, routing hints, hints lookup service and forward lookup service are explained in detail.

3.3.1 Naming Format

```
ni://example.com/sha-256;30jRPW_2jpc04Fq8ifw3cU3EaRBfm8qqlsUd_K0_DBg
```

Figure 3.1: sample NDO name

As stated earlier, NetInf NDO names use ni URI scheme [24]. ni names are flat and supports name data integrity by including a cryptographic hash of the NDO. A sample NDO name is shown in Figure 3.1. In the sample NDO name, *ni* denotes the naming scheme. NDO names include optional authority part that can be used by applications to access corresponding NDOs. Authority part in the above sample is *example.com*. ni URI scheme allows using various digest algorithms to generate hashes. The name of the digest algorithm used can be identified from the name itself. Digest algorithm sha-256 is used in the above NDO name. The final part of NDO name is the cryptographically generated

hash using NDO itself. The hash value can be used to verify the integrity of a NDO. Apart from these parameters, additional fields can be added to ni URI by using “tag=value” list. There are various naming schemes used by different ICN approaches. The global routing solution can work with any naming scheme that includes authority part or other component that specifies a group of NDOs, for example, the above mentioned scopes..

3.3.2 Hint Lookup

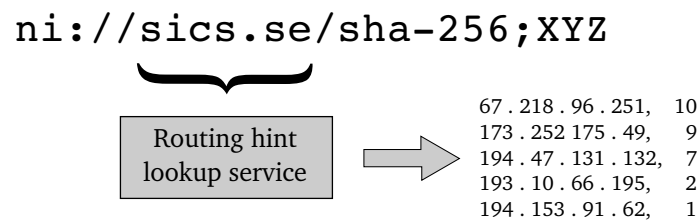


Figure 3.2: Routing hint lookup

The authority part of the NDO name contains a domain name. Routing hint lookup service maps the domain name to one or more routing hints each associated with a priority value as shown in Figure 3.2. The routing hints are used to route NDO requests towards the NDOs’s location. It is the responsibility of content publishers to make domain names to routing hints mapping available in the global ICN network. The existing DNS system can be extended to provide the hint lookup service. In addition, local hint lookup table can be maintained at few core routers in order to route GET requests towards more specific location.

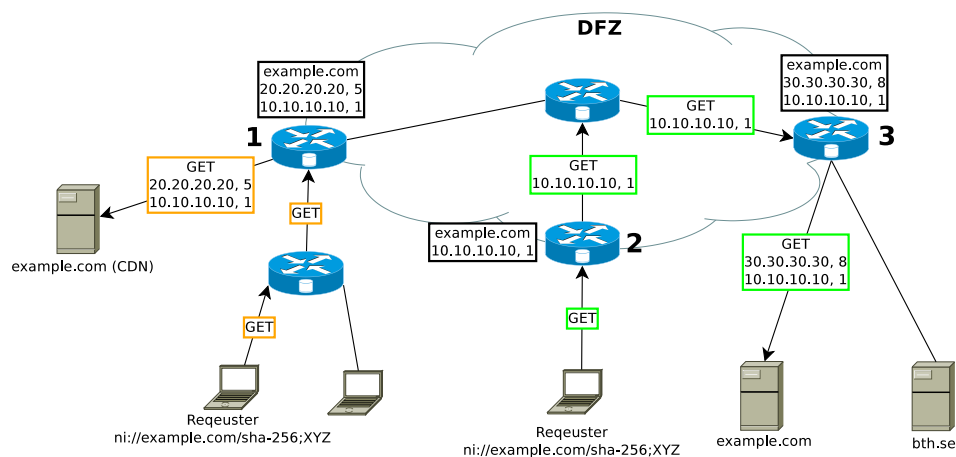


Figure 3.3: Hint lookup in edge network and DFZ

Hint lookup is first performed when a NDO request is routed from edge network to DFZ. Figure 3.3 shows three edge routers that perform hint lookup. Hint lookup at router 2 produces a low priority routing hint (10.10.10.10, 1) that is globally distributed. The low priority routing hint is then used in DFZ routers to forward GET request towards server location. A different usecase of hint lookup is performed at router 1. In addition to globally distributed routing hint (10.10.10.10,1), router provides a higher priority routing hint (20.20.20.20, 5). The high priority routing hint enables GET request to be forwarded to nearby CDN rather than DFZ. Finally hint lookup is also performed at router 3 that is located at server edge network. Routing hint (30.30.30.30, 8) is typically used to route GET request towards appropriate node in server edge network.

```
ext={"hint":["30.0.0.1,10","40.0.0.1,8"]}
```

Figure 3.4: Sample ext field with routing hints

In any case, newly retrieved routing hints are added or merged to the GET request before forwarding to nexthop node. As per NetInf protocol definition, GET request includes an *ext* field to handle future extensibility of NetInf protocol. This field is in json coded string format. Routing hints are encoded as part of this *ext* field. A sample ext field with routing hints is shown in Figure 3.4. Routing hints are inserted as a JavaScript Object Notation (JSON) attribute that is named *hint*. The value of the attribute is a list of string values. Each of the string values contains one routing hint and its priority value separated by a comma sign. Routing hints and priorities are discussed in more detail in the following section.

3.3.3 Routing Hints

As stated earlier, routing hints are used to route GET requests towards NDO's location. Routing hints are typically used when a router does not have a requested NDO in its cache and it does not know exact location of a requested NDO. Explicit aggregation is performed on routing hints each associated with a priority value.

Low priority routing hints sits at top level of aggregation that are distributed all over the global network. A GET request's initial direction can be set using low priority routing hints. As GET request follows the directed path, following routers can add higher priority hints to GET request. Hence, GET requests can be routed towards more specific location. For instance, GET request can be routed to CDN caches or to a specific server in publisher network. Routing hint

uses IPv4 namespace however, there is no implicit aggregation in the name structure itself. Routing hints follow flat namespace hence, each IPv4 address is different and serves the same purpose.

3.3.4 Forward Lookup

A forward lookup in NetInf router is done in order to retrieve nexthop address. GET request can be forwarded to the retrieved nexthop address using underlying transport protocols. A NetInf router can have two forwarding tables, they are name forwarding table and routing hints forwarding table. The name forwarding table provides a mapping from NDO name to nexthop location and this table is primarily intended for edge networks.

The routing hints forwarding table contains the mapping from the routing hints to nexthop addresses. As stated earlier, routing hints use the IPv4 namespace, but the routing hint does not refer to any network location. Hence, the present IP routing mechanisms can be used to populate the routing hints forwarding table. The lookup is performed using exact match and GET request is routed to the resulting nexthop address. A GET request can contain multiple routing hints. The lookup into the routing hints forwarding table is performed from high priority hint to low priority hint until an exact match is found. An example routing hints forwarding table is shown in table 3.1. The nexthop addresses denote NetInf enabled routers or end points, that can either route GET requests or respond to GET requests.

Routing hint	Nexthop Address
10.0.0.1	http://sics.se/netinfproto/get
20.0.0.1	http://ericsson.com/netinfproto/get
30.0.0.1	http://orange.com/netinfproto/get

Table 3.1: Locator forwarding table

3.4 Forwarding Steps

NetInf global routing includes numerous steps that are executed when a router forwards a GET request. These steps are given below.

- Receive GET request and extract routing hints if present.
- Cache lookup: The requested NDO's name is looked up in local cache. If found, respond to requester.
- Hint lookup: Lookup authority part of the NDO name in the hint lookup table. If found, add or merge retrieved routing hints to GET request.
- Forward lookup: The routing hints are looked up in the forward lookup table for an exact match. If found, retrieve nexthop location of highest priority routing hint with exact match.
- Forward the GET request to the identified nexthop location.

This chapter presents the NetInf Global routing implementation in detail. NetInf global routing is implemented as part of NEC's NetInf prototype which is called NEC NetInf Router Platform (NNRP) [27]. After briefly explaining NNRP architecture, the implementation of global routing modules are presented in detail. Finally, source code licensing information of the newly added global routing modules is stated.

4.1 Build Methodology

Since the NetInf routing solution is added to the NNRP software, the NNRP software design is well understood first. The optimal way to integrate the routing solution into the NNRP is then figured out. The interfaces are defined for the communication between the existing modules and the new modules. The inbuilt NNRP libraries and few opensource libraries are used in the new modules. The new modules are developed using the programming language that was used to develop the NNRP software. A simple automation test suite is developed to verify the routing of GET requests. The implementation is tested simultaneously using the automation test suite during the development phase. The new modules are well documented as part of the NNRP documentation. Subversion is used as the version control system for the new source code.

4.2 NEC NetInf Router Platform

NNRP implements various NetInf node functions such as, ni URI naming scheme, HTTP convergence layer, NDO cache, support for publishing NDOs, and name resolution. NNRP is intended to run on POSIX compatible operating systems. NNRP was developed using the standard C programming language. NNRP follows a modular and extensible software architecture. Hence, it provides a flexible environment to develop ICN networks and testbeds. NetInf messages are processed through chains of modules in NNRP. The chains of modules can be configured using a simple configuration file. A NetInf message

can be received through different convergence layer modules. A received message can be conditionally moved and processed across different chains of modules. A processed GET message can be easily forwarded to nexthop and GET-RESP can be forwarded back to requestor over a convergence layer.

4.2.1 Message Structure

The NNRP message structure is shown in Figure 4.1. The type field indicates NetInf message type. The name field contains NDO's name which is basically a ni URI. The metadata part of the message stores a collection of key value pairs. Each key value pair attaches a state into processed message.

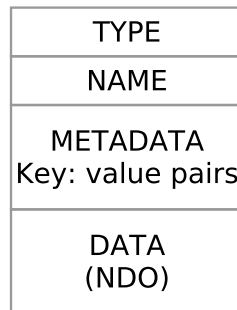


Figure 4.1: NNRP message Structure

4.2.2 Basic Concepts

Modules, instances and chains are the fundamental elements in NNRP. Modules execute various set of actions and the modules can also interact with each other. In NNRP, the modules are instantiated in order to execute their actions. An instantiation of a module is known as an instance. Each instance has a unique instance id. The behavior of an instance can be influenced by instance specific configuration parameters. A sample arrangement of chains and modules is shown in Figure 4.2. The arrangement contains three chains namely input, process and output. As the name states, the input chain receives a NetInf message, performs preliminary checks and feeds the message to the process chain. The process chain consists of modules that perform core NetInf functions. Also, the process chain is responsible for forwarding GET requests to nexthop locations and receiving the corresponding GET-RESP messages from the nexthop locations. Finally, the process chain moves the possibly transformed message to the output chain. The output chain's primary responsibility is to send the transformed message back to the previous NetInf node.

A module can at most provide three different functions that can be applied on a message. These functions are referred to as source, filter and sink. The source function of a module feeds a new message into the system. As shown in figure 4.2, the source function of the convergence layer module is located at the beginning of the input chain. The source function can receive a NetInf message from the network and feed it to the chain. A filter receives a message and may return a transformed message. For example, the filter function of the cache module can lookup the local cache on processing a GET request. If the requested NDO is found in the cache, the GET request can be transformed to the corresponding GET-RESP message. Otherwise, the GET request will just be passed to the next module. A sink function is similar to a filter. However, a message processed in a sink can either be moved to a different chain or just dropped. The sink function of a convergence layer is located at the end of the output chain in the sample NNRP configuration that is shown in Figure 4.2.

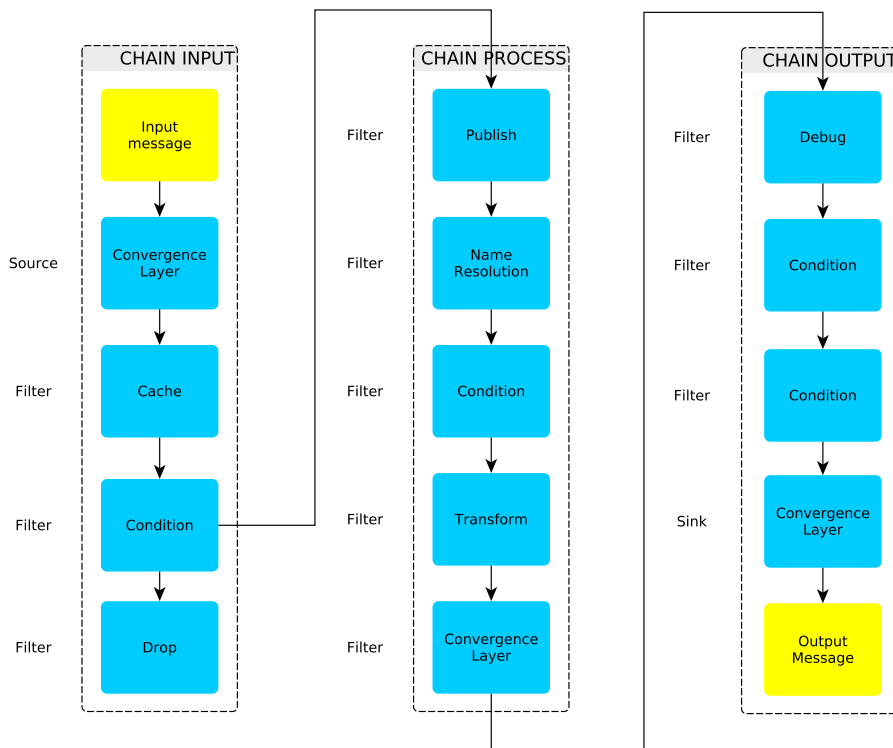


Figure 4.2: NNRP chains and modules

This sink function sends the NetInf message to an external NetInf node and thus terminates message processing at the local NNRP node. As shown in figure 4.2, chains are formed by sequence of filters and optionally including source and sink objects. In addition, NNRP provides flexibility to move or copy messages

across chains. The filter function of a condition module can be used to perform an action if a condition is met. For example, a message can be moved to the process chain if the message type is GET.

4.3 NetInf Global Routing

NetInf global routing introduces two new modules into NNRP. The new modules are known as hint lookup module and forward lookup module. The two modules are referred as the global routing modules. The global routing modules receive GET requests, optionally add global routing parameters into metadata field and passes GET requests to the next module. The global routing modules do not generate or terminate messages, but add metadata into messages. The global routing modules provide only the filter function. The source and filter functions of the HTTP convergence layer module are modified to support NetInf global routing. The processing sequence of GET request forwarding that include NetInf global routing in NNRP is shown in Figure 4.3.

4.3.1 Processing Sequence

The source function of HTTP convergence layer is placed at the beginning of the input chain. The source function receives a NetInf message from the network and feeds it to the next module. The next module checks if the received message's type is GET. The message is moved to the process chain if the condition is met. Any messages that are not of type GET is passed to the drop module and eventually gets dropped.

The process chain receives GET request from the input chain, where the message is first fed to cache module. The cache module checks if the requested NDO is present in the local cache. If present, the GET request will be transformed to GET-RESP and the requested NDO will be added to the message. The name parsing module parses the NDO name that is in ni URI format. Authority part, hash function name and hash string are extracted from the NDO name. The extracted values are added to the metadata field of the GET request so that they can be used in the following modules. The GET request is then passed to the hint lookup module. The hint lookup module fetches routing hints for the extracted authority part in NDO's name. The GET request is then forwarded to the forward lookup module. The forward lookup module looks up the given routing hints and sets the resulting nexthop address in the GET request. Internal processing in the global routing modules is explained in detail later in this chapter.

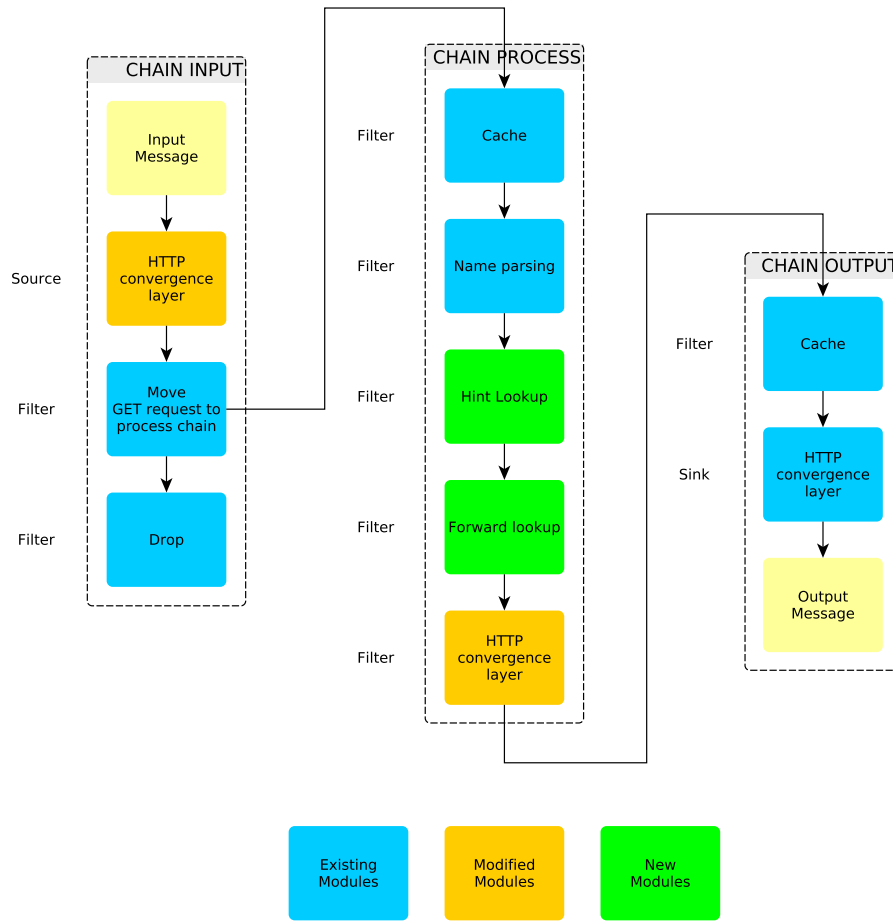


Figure 4.3: NNRP NetInf global routing

The filter function of HTTP convergence layer is located at the end of the process chain. The GET request received by the convergence layer module is forwarded to nexthop node. Once the corresponding response is received from nexthop node, the filter function moves the response to the output chain. Let us assume that the received response is of type GET-RESP and contains the requested NDO. The received GET-RESP will be moved to the output chain. As shown in figure 4.3, the output chain contains cache module and http convergence layer module. GET-RESP is first fed to the cache module in the output chain. On receiving GET-RESP, the cache module adds the received NDO into the local cache, so that the following GET requests to the same NDO can be served from the local cache itself. Finally, the GET-RESP is fed to the sink function of HTTP convergence layer module. This sink function checks if the message type is GET-RESP. If so, the response will be sent back to the previous NetInf node from which the GET request was received.

4.3.2 HTTP Convergence Layer Module

As stated in the previous chapter, NetInf global routing solution adds routing hints to GET request. Routing hints are inserted to GET request as shown in figure 3.4. Hence, HTTP convergence layer module in NNRP is modified in order to insert and extract routing hints.

source function

The source function of HTTP convergence layer module, retrieves JSON coded string that comes as *ext* field value of the GET request. If *ext* field contains routing hints, the hints are extracted along with priority values. The extracted hints and priority values are converted from text format to binary format. The hints and priority values in binary format are inserted into a linked list and the list is added to the metadata field of the GET request. Insertion sorting is performed to keep the list starting with high priority hint and ending with low priority hint. The routing hints and the priority values are then used in the global routing modules.

filter function

The filter function of HTTP convergence layer module forwards the GET request to the nexthop node. The filter function is modified to insert the sorted list of routing hints and priority values into the forwarded GET request. The hints and priority values are converted from binary format to text format. The resulting text values are added to JSON coded string as *ext* field value of the forwarded GET request. NetInf nodes in the routed path can use the routing hints list in GET request to route further.

4.3.3 Hint Lookup Module

The hint lookup module maps the authority part of the NDO name into routing hints. Apart from the routing hints extracted in the source function of HTTP convergence layer module, the hint lookup module can optionally provide additional routing hints for a given authority part. The additional routing hints are merged to the routing hints extracted in the source function and the resulting routing hint list is sorted. The hint lookup module provides only the filter function. The mapping can either be done using a local hash table lookup or DNS lookup. The internal processing flow of hint lookup module is shown in Figure 4.4. Hint lookup module first checks for authority part of ni name. If authority part is not found, GET request is just passed to the next module in chain. The hint lookup table in local NNRP node is implemented as hash table. Each entry in the hash table contains a pointer to linked list of routing hints.

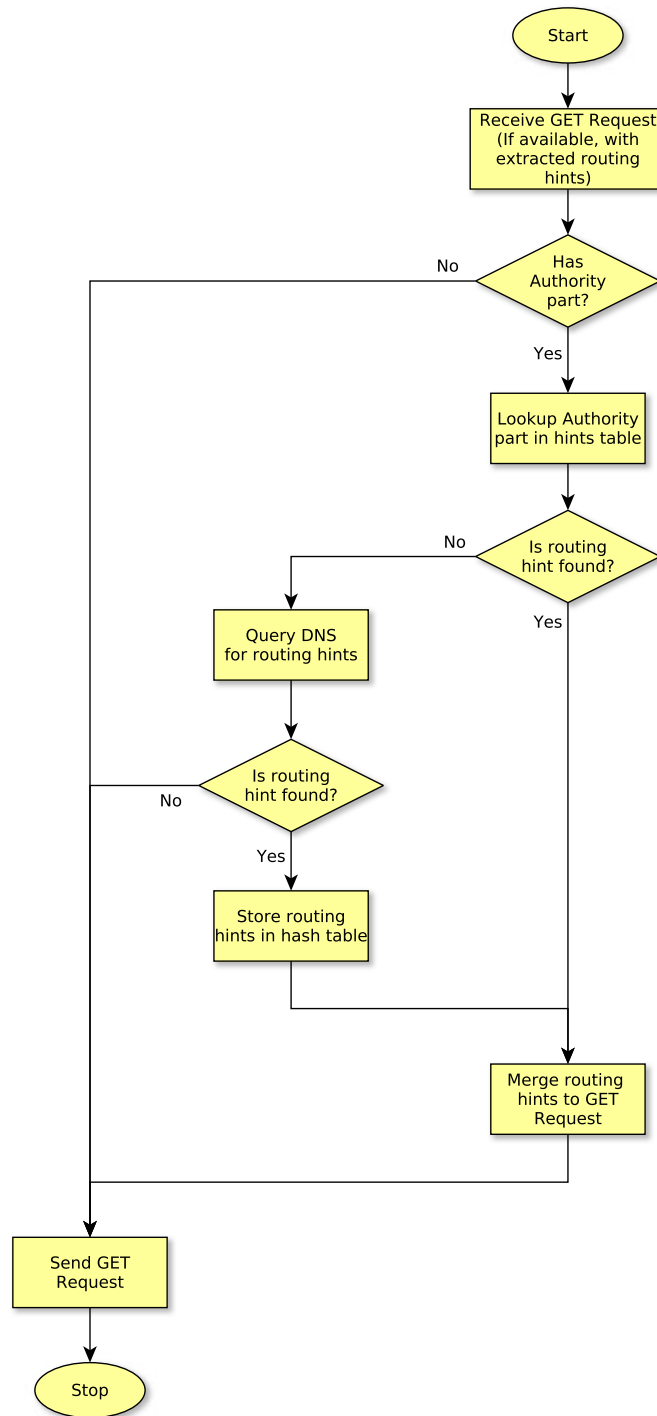


Figure 4.4: Hint lookup processing flow

The authority part of the ni name is used as key in routing hints hash table. If there is a corresponding entry in the hash table for a given authority part, a linked list of routing hints is retrieved. If no entry is found in the hash table, extended DNS lookup is performed to retrieve routing hints. DNS record of type TXT is used to serve routing hints. A DNS query for TXT type records is sent for a given authority part or the so called domain name. Each TXT record in the resulting DNS query response may contain a routing hint and priority value separated by comma sign. Routing hints retrieved in a DNS query response will be added to the local hash table. Hence, routing hints can be found in the local table for the following GET requests with the same authority part.

The retrieved routing hints are added to the GET request. In order to avoid redundant routing hints in a GET request, a routing hint will be inserted only if it is not already found in the GET request. Insertion sort is used in order to keep the routing hint list sorted. The merged linked list of routing hints is added to the metadata field in the GET request. This list will be used in the forward lookup module in order to find the nexthop location.

4.3.4 Forward Lookup Module

The forward lookup module optionally sets the nexthop location to which the GET request will be forwarded. The forward lookup module provides only the filter function. The forward lookup module contains two tables, namely a forward hint table and a forward nexthop table. The forward hint table is a hash table that stores mappings from routing hints to index values. The forward nexthop table contains mappings from the index values to a routing hint, nexthop address and nexthop port number. The routing hint is associated with a physical nexthop location at each NetInf node. The nexthop address and port number in the entry refers to associated nexthop's http location. In principle, numerous forward hint table entries may refer to the same forward nexthop table entry. Since only index to forward nexthop table is stored in forward hint table, redundant storage of nexthop information is optimized.

The processing flow inside the forward lookup module is shown in Figure 4.5. The filter function of the forward lookup module first checks if the received packet contains a list of routing hints. The list can include hints added by the local hint lookup module or hints that are received in the GET request. Also, the list contains routing hints in descending order of hint priority. Hence, high priority hints are processed first. Routing hints from the list are looked up in the forward hint table until a match is found.

Since the lookup is carried out in the same sorted order, the lookup will return a match for the routing hint with highest possible priority. In case a match is found, an index to the forward nexthop table will be retrieved. If no match is found, the filter function will check if there is a default nexthop configuration. A default nexthop configuration is just an index to forward nexthop table. The corresponding nexthop address will be retrieved from the forward nexthop table. The retrieved address will be set in the NNRP internal state of the GET request, so that the GET request will eventually be forwarded to the retrieved address. The GET request that is optionally set with a nexthop address will then be forwarded to the next module in NNRP.

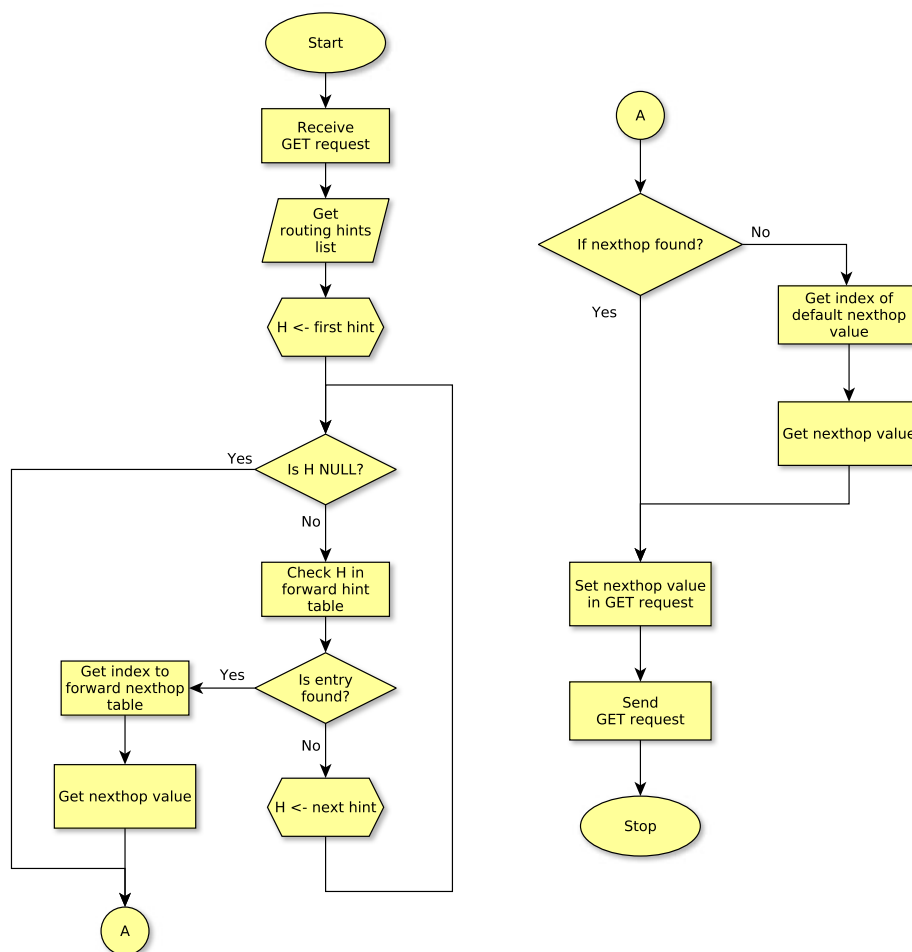


Figure 4.5: Forward lookup processing flow

4.4 Licensing

The source code of the newly added NetInf global routing modules are developed as an open source. The source code is licensed under Apache license version 2 [28]. Yogaraj Baskaravel, SICS Swedish ICT AB, Sweden and Blekinge Institute of Technology, Sweden are the copyright owners of the source code.

4.5 Observation

Certain challenges in implementing NetInf global routing have been identified. The routing hints and the associated priority values are added as part of the *ext* field in the NetInf message. The *ext* field uses JSON encoding, and therefore also does routing hints and their priorities. The routing hint and priority values are converted to binary format for processing inside NNRP. The binary data is then converted back to JSON in ASCII format while forwarding the GET request. Hence, ASCII parsing adds up unwanted processing overhead at each node. The GET request message format can be redefined to include binary data objects, so that routing hints and priority values can be transmitted more efficiently.

On receiving a GET request message, NNRP parses and extracts all the message contents. The extracted data is then stored in an internal message structure and the received message is discarded. The internal message structure is used for processing inside NNRP modules. While forwarding the GET request to the next hop node a new GET request is framed and sent using the data stored in the internal message structure. Instead, a received GET request can be just forwarded after doing the required modifications. Further experiments and analysis should be done on how to handle a received GET request at each node.

This chapter presents the experiment design and the detailed analysis of performance measurements. The experiment was designed to capture the time taken by various steps in forwarding a GET request. The time taken by various steps in the GET request forwarding are discussed in detail. The time measurements were logged in such a way that there was minimal disturbance on actual measurements.

5.1 Experiment Design

The experiment was carried in order to gain further insights into the time taken by various steps in the GET request forwarding. Each GET request that passed to a particular router in the experiment setup is considered as a subject. The time measurements and the received hints count are logged for each GET request in the measured router. The time taken by various steps and number of routing hints in the received GET request forwarding are the dependent variables. The nexthop address to which the GET request is forwarded is an independent variable. The forwarding time will vary with respect to the number of routing hints in the received GET request. The GET requests are randomly inserted with one to five number of routing hints. Random assignment technique is used to send GET requests with different number of routing hints to the measured router. The experiment follows a within-subjects design such that the variance is controlled by changing number of routing hints in the GET request.

5.1.1 Experiment Setup

The experiment was carried out in the NetInf testbed that was built on top of the Internet as shown in Figure 5.1. The NetInf testbed nodes were placed across three SAIL partner sites. They are SICS, Ericsson and France Telecom. The nodes were setup to run NNRP with the NetInf global routing implementation. Apart from the NNRP nodes, requester nodes were placed at all three partner sites. Numerous NDOs were published from all the three sites prior to the measurements. The end nodes were automated to send GET

requests to retrieve NDOs from the remote locations repeatedly. The end nodes were automated to vary the number of routing hints in the GET requests sent. The GET requests were sent in a sequential manner such that a new request is sent as soon as the current request is served. Certain steps in the GET request forwarding are affected by the number of routing hints in the received GET request. The number of routing hints in the received GET request is logged as part of the measurement. Hence, the time taken by the relevant steps are discussed with respect to the number of hints received. The received routing hints are merged into the hints found in the hint lookup step. Insertion sort is performed while merging the newly found routing hints to the received routing hints. Hence the time taken in this step is affected by the amount of sorting computations. Since routing is performed only after a cache miss, the cache was always kept empty during the experiment. Hence, the cache lookup always resulted in a miss.

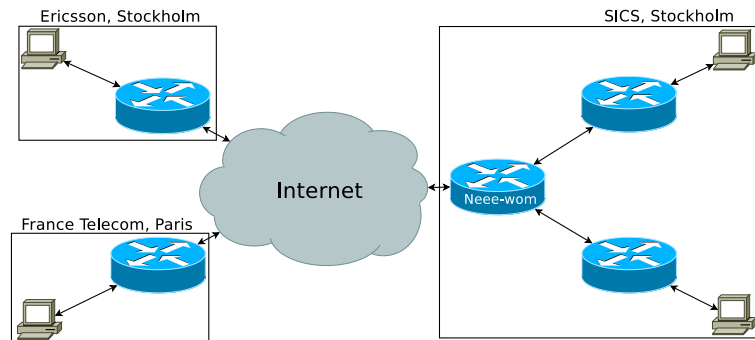


Figure 5.1: Experiment NetInf Testbed

5.1.2 Measured Node

In order to have a mixture of routing to nearby and remote nodes, measurements were taken in the gateway node at SICS. The gateway node at SICS is named as Neee-wom. Table 5.1 presents Neee-wom's general system information. As shown in Figure 5.1, Neee-wom was connected to two nodes at SICS and two remote nodes. Remote nodes were located at Ericsson, Stockholm and France Telecom, Paris, one at each site respectively.

Timestamps were collected at various points of forwarding GET request in NNRP. The timestamps were used to measure the processing time of each individual step. The timestamps were accessed from raw hardware clock in order to avoid any dynamic changes in clock that might affect the measurements. The timestamps were dumped into a file in such a way that there was minimal disturbance on actual measurements. At the end of forwarding each GET request, the timestamps of the GET request were added

Operating system	Ubuntu 12.04.1 LTS
Ethernet controller	Intel Corporation 82579LM Gigabit Network Connection
Architecture	x86_64
Processor	Intel(R) Core(TM) i5-3570 CPU @ 3.40GHz
Memory	4GB

Table 5.1: Nee-wom system information

to a global data pool. The data from the global data pool was then periodically dumped to file by a separate thread that was running as part of the NNRP process. Since the dumping is handled by a separate thread, the routing thread does not have to wait until timestamps are dumped into the file. Hence, the time taken for logging the measurements was reduced as much as possible such that the impact on measured metrics was very low.

5.1.3 Pilot Study

A pilot study was carried before the actual experiment in order to validate the data collection. The transmission of GET requests with one to five routing hints in random assigned manner from the end nodes were tested. The gateway router at SICS is verified that the router properly dumps the time stamps property. The content of the measurement dump file is validated. A brief analysis was carried out on the time taken in various steps of the GET request forwarding.

5.1.4 Validity Threats

The process scheduling in the nee-wom machine could affect the measured time values. In case if the processor was scheduled to run some other application or system process while GET request is being processed, the NNRP process would have to wait. The waiting time will be accounted as part of the GET request forwarding incorrectly that will lead to an internal validity threat. In order to avoid this to a certain extent, no user applications were run in the nee-wom during the experiment. An external validity threat is that the round trip time to different nexthop nodes were different, since the nexthop nodes were located at varying distances. However, the experiment aims to capture the variance in GET request forwarding corresponding to the distance to the nexthop node. The measurements corresponds to the NNRP NetInf global routing arrangement shown in figure 4.3. This limitation relates to a construct validity threat.

5.2 Performance Evaluation

Analysis of the overall forwarding time is first presented in this section. An overall picture of the time taken in each step in GET request forwarding is then presented. After that, the time taken in each forwarding step is discussed in detail.

5.2.1 Statistical Significance

The experiment setup was running for around two days and measurement dumps were collected. The measurements were collected for a total number of 224,529 GET requests. The time measurements for each step in the GET request forwarding is evaluated in detail. The various time values in the measurements are sorted in ascending order and then all the outliers are removed before the evaluation. In order to avoid that the analyzed time value could have occurred due to chance, the median values of the time measurements are used in all the following performance evaluations.

5.2.2 Overall Forwarding Time

The frequency distribution of the overall forwarding time is presented in Figure 5.2. The figure clearly shows that the measurements fall into three value ranges. The global routing node creates a new connection with the nexthop node to forward each GET request. The cumulative distribution of the overall forwarding time and the time taken to establish the nexthop connection are shown in Figure 5.3. As Figure 5.3 shows, the time taken to establish a nexthop connection is slightly lower than the overall forwarding time. Hence, the three value ranges are caused by the time taken to establish the connection to the nexthop.

As the connection establishment time varies with respect to the distance to the nexthop node, the three ranges should relate to the connection establishment from Nee-wom to 1. SICS nodes, 2. Ericsson node and 3. France Telecom node. The routing was implemented using the HTTP convergence layer approach resulting in that NNRP establishes a new Transmission Control Protocol (TCP) session to forward each NetInf GET request. The session establishment takes a highest amount time among the other steps in the GET request forwarding. Connection establishment is not part of the NetInf global routing solution. However, HTTP connection type needs to be specified for the NetInf HTTP convergence layer approach. The TCP session establishment overhead for forwarding each GET request can be avoided by using HTTP keep-alive connection. Hence, many GET requests destined to the same nexthop node can be sent by reusing the same TCP connection.

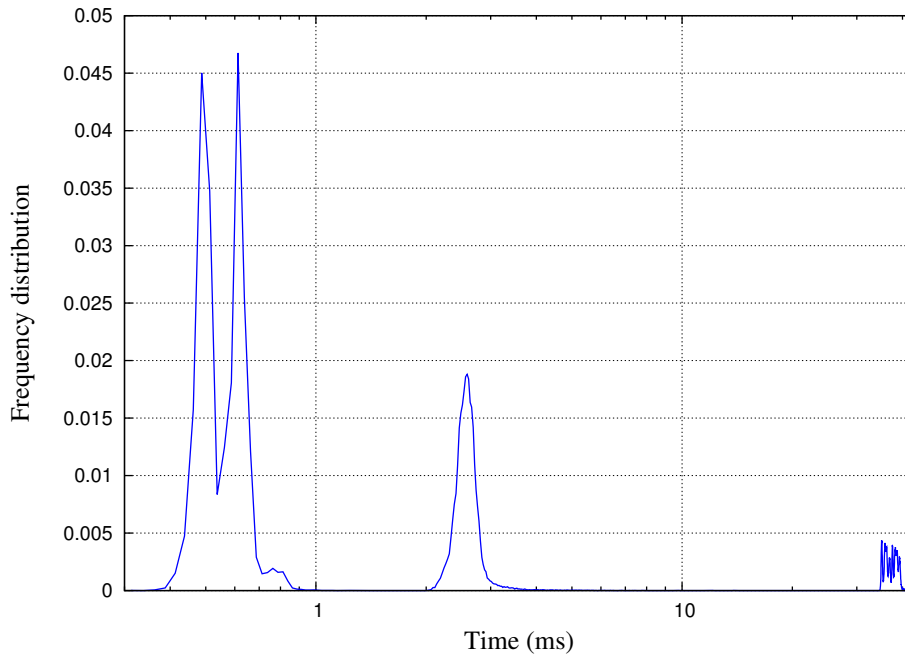


Figure 5.2: Frequency distribution of overall forwarding time

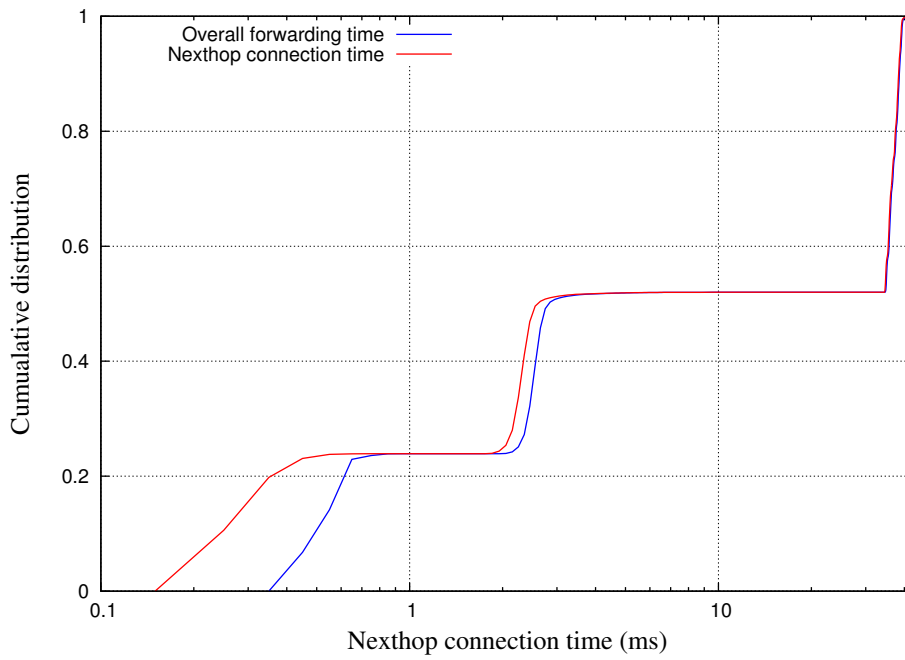


Figure 5.3: Overall forwarding time vs nexthop connection time

Step	Description
General extract	Extracts NDO's name and msg id from GET request
Hints extract	Extracts routing hints from GET request
Parse	Classifies request type, optionally performs cache lookup and parses NDO's name
Hint lookup	Looks up and adds routing hints for a given authority part
Forward lookup	Looks up and sets nexthop address using given routing hints
Connect nexthop	Establishes connection with nexthop node
Frame forward	Frames and forwards GET requests to nexthop node

Table 5.2: NetInf global routing steps

5.2.3 Forwarding Steps

A brief description of the steps in GET request forwarding is given in the table 5.2. The median time taken by each step in NetInf global routing is presented in Figure 5.4. The global routing steps are discussed after presenting the steps that perform usual NetInf functions. The time taken by the general extract step lesser than the time taken by the other steps in GET request forwarding. The parse step takes around $135\mu s$. The parse step involves taking a lock, adding the GET request to a global queue and spawning a new thread to continue processing the received GET request.

The difference between the yellow bar and the blue bar is the time taken to perform a cache lookup. The cache lookup time is comparatively lesser than the other steps as the cache was kept empty. The cache lookup enables router caches to be utilized very well. As discussed earlier, the connect nexthop step constitutes a major portion of the overall forwarding time. The time taken by the connect nexthop step is affected by the distance between Nee-wom and the nexthop node and TCP responsiveness of the nexthop node. The above three steps do not relate to NetInf global routing. Hence, the time taken by these steps are not affected by the NNRP global routing modules. Measurements of steps that relate to NetInf global routing are discussed below.

When forward lookup is performed, the forward hint table is looked up first. If a matching entry is found, for the corresponding nexthop address will be set for the GET request. In case if a match is not found, the default nexthop address will be set for the GET request. In Figure 5.4, the green bar denotes the time taken to lookup the nexthop address in the forward hint table. Whereas the

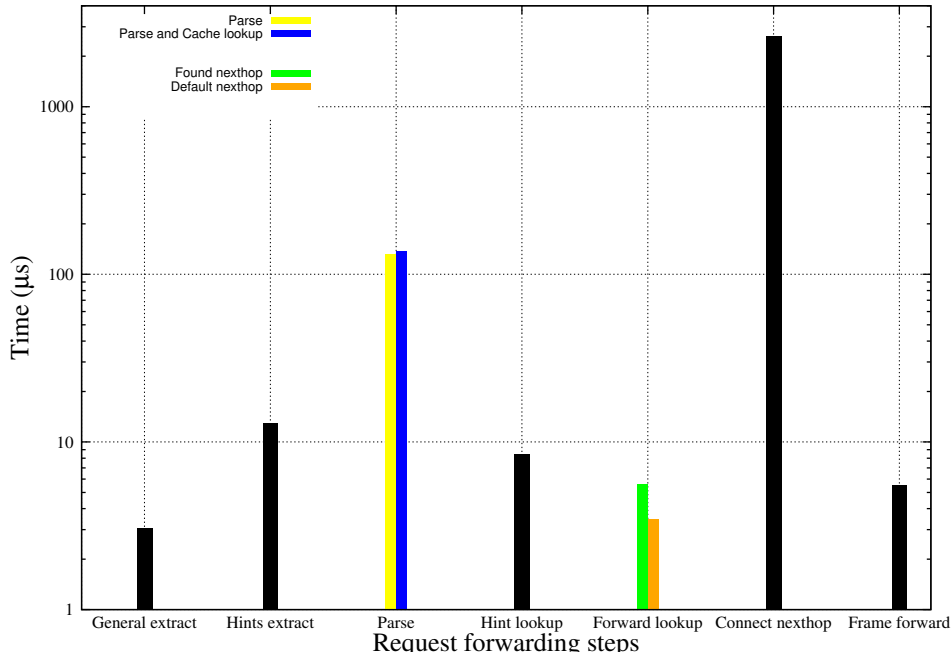


Figure 5.4: Median time taken by GET request forwarding steps

orange bar denotes the additional time taken to retrieve the default nexthop address. Retrieving the default nexthop entry involves accessing a global variable and a global array which are shared among multiple threads. The median time taken to retrieve the default nexthop address is $3.44\mu s$. However, GET requests with no routing hint or unknown authority part can still be forwarded. The hint and forward lookup steps are part of NetInf global routing. The hint extract and frame forward steps were modified to extract and insert routing hints respectively. Each of these steps is explained in detail in the following sections.

5.2.4 Extracting Hints

As the first routing step, routing hints are extracted from a received GET request. The end nodes used in the experiment setup were configured to include no routing hints or one to five number of routing hints in a random manner. Hence, a received GET request may or may not have routing hints in it. Figure 5.5 shows the time taken to extract varying number of routing hints along with the standard deviation. As shown in Figure 5.5, even if there is no routing hint in a received GET request, the median time taken to parse the empty JSON structure is $1.37\mu s$. The time taken to extract routing hints gradually increases as the number of hints extracted varies from one to five. However, an initial setup time is commonly present in all the five data points which could be the time to

create the base data structure. JSON parsing involves intensive computation, since it might include different kinds of JSON fields. The hint extract step takes a notable amount of time which can be optimized further. The GET request format can be redefined to include routing hints and priority values in a better way, so that unnecessary overhead incurred in text parsing and JSON parsing can be avoided.

5.2.5 Hint Lookup

Nee-wom was configured to perform hint lookup irrespective of routing hints present in a received GET request. Hint lookup is basically a hash table lookup that returns a pointer to the list of routing hints. Hence the time taken to retrieve varying number of routing hints take approximately the same time. Figure 5.6 shows the time to retrieve one to five number hints. The time taken to perform hint lookup is more or less similar to the most of the other forwarding steps.

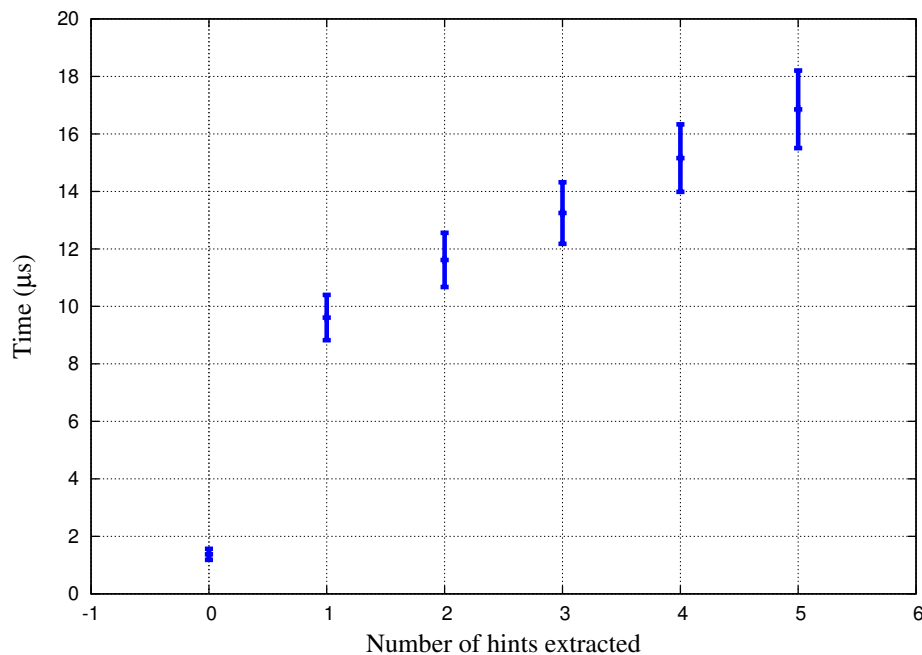


Figure 5.5: Time taken to extract hints

5.2.6 Merging Hints

The hint lookup step shown in Figure 5.4 also includes merging routing hints. The hints that are the result from the local hint lookup will be merged with the hints that are extracted from the GET request. The extracted hints are sorted as they are extracted from the GET request. Routing hints that are stored in the local table is also sorted. Sorting is done in order to have higher

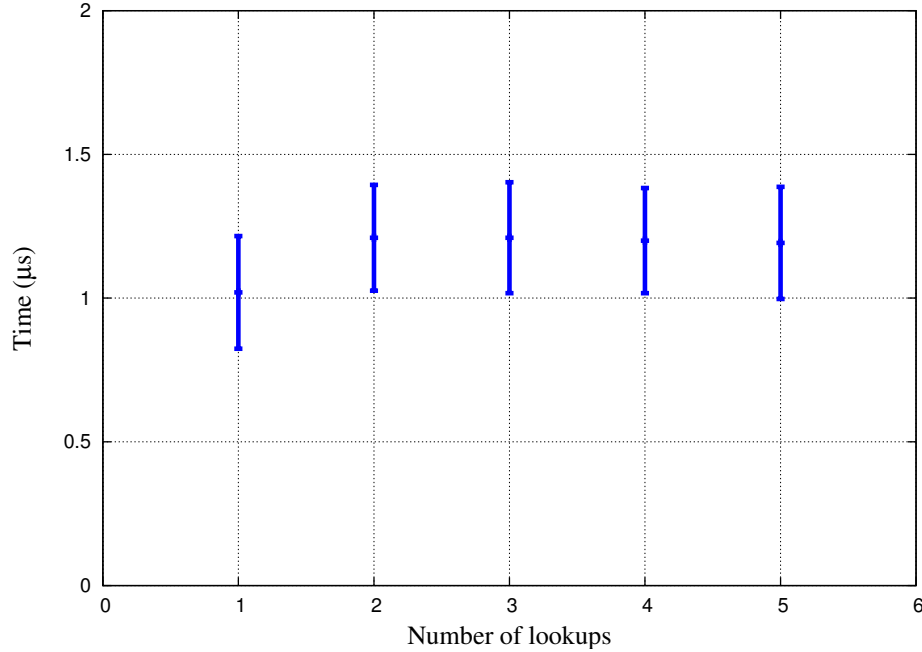


Figure 5.6: Time taken to perform hint lookup

priority hints in the beginning. The cumulative distributions of the time taken to merge one hint to five hints are presented in Figure 5.7. An interesting observation is done in case of merging just one new routing hint. Almost half of the measurements that corresponds to merging one routing hint is approximately equal to $2\mu\text{s}$. The value is notably less than the rest of the measurements. Hint merging step basically merges one sorted list into another sorted list. Hence, less number of new hints produces less number of comparisons and eventually takes lesser time.

In the experiment setup, most of the GET requests with one hint had higher priority hint. Since newer routing hints had comparatively lower priority, just one comparison is needed for merging and this eventually resulted in lower time. As shown in Figure 5.7, the distribution of measurements that falls around $8\mu\text{s}$ is very low. The low distribution should also be the result of fewer comparisons while merging hints. The hint merging implementation dynamically allocates and frees memory. Dynamic memory allocation and freeing takes mutual exclusion locks to be thread safe [29]. Based on GET requests inflow, multiple hint merging would have been carried out simultaneously by using multithreading in the global routing implementation. Hence, dynamic memory usage in hint merging can be optimized further.

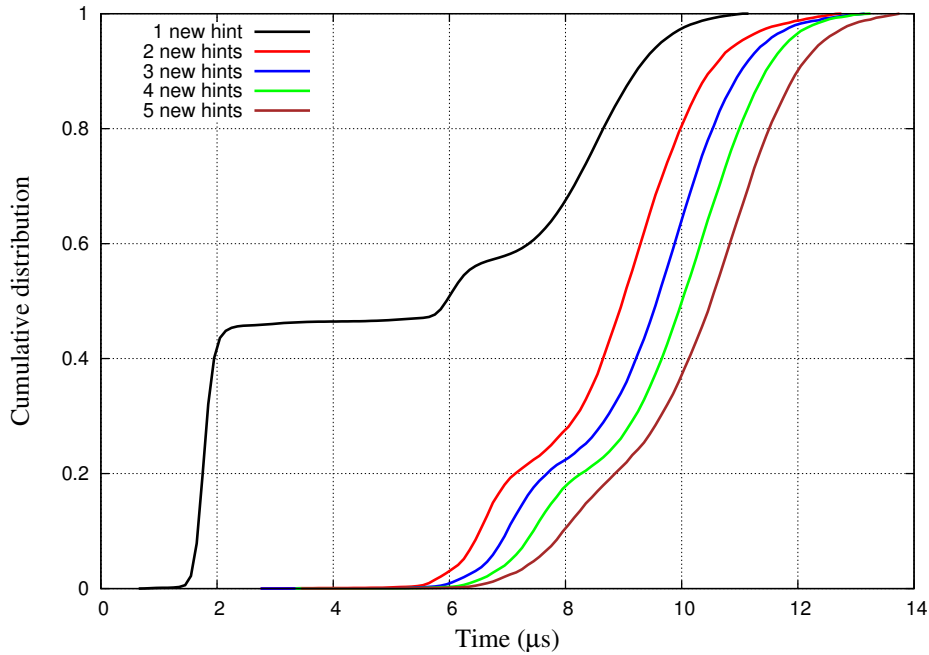


Figure 5.7: Time taken to merge new routing hints to existing routing hints

5.2.7 Forward Lookup

The GET request along with the merged list of routing hints is then processed by the forward lookup step. The forward lookup is done for each routing hint in the list, until an exact match is found. The list contains hints in descending order of hint priority. Hence, the forward lookup will return exact match of the hint with highest priority. The nexthop address is retrieved from the matching entry and is added to the GET request. Figure 5.8 shows the time taken to perform the forward lookup. The time taken to perform the forward lookup slightly increases with respect to the increase in number of lookups. The major part of this time is possibly spent in accessing the forward lookup table and adding the nexthop address. Overall, the time taken by forward lookup step is less than the total time taken by hint lookup and hint merging.

5.2.8 Frame Forward

Finally, NNRP frames a new GET request from the internal NNRP message structure. The new GET request is then forwarded to the nexthop address over the HTTP convergence layer. In terms of NetInf global routing, routing hints and priority values are inserted into the framed GET request. Figure 5.9 shows the cumulative distributions of the time taken to frame and forward GET requests for varying number of routing hints insertion. This step involves

communication over the underlying HTTP convergence layer. The HTTP convergence layer approach is good enough for a research prototype, but the networking stack that sits between the network interface and the HTTP layer will also add some delay. Since the implementation uses an HTTP connection, some time is spent in waiting for TCP ACK packets. Also, the implementation performs dynamic memory allocation while inserting each routing hint to the GET request.

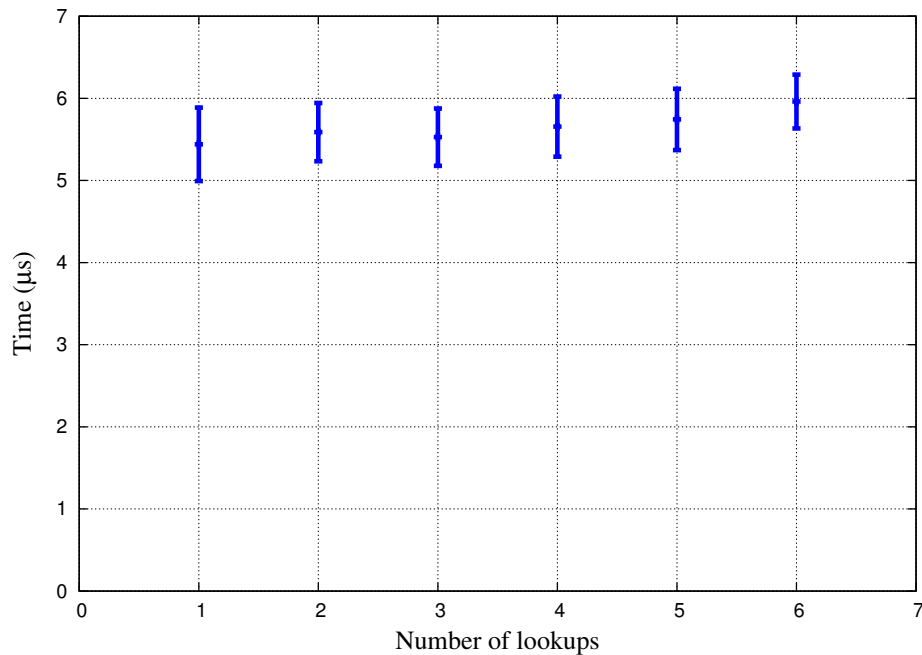


Figure 5.8: Time taken to perform forward lookup

5.2.9 DNS Hint Lookup

Apart from performing hint lookup locally, the implementation also supports hint lookup from a DNS server. Whenever local hint lookup fails, routing hints are retrieved from the DNS. The time taken to retrieve hints from a DNS server is shown in Figure 5.10. The median value of DNS hint lookup time is shown for one to five routing hints. If routing hints are found in DNS, the response to a DNS hint lookup query includes routing hints and their priorities.

In the experiment setup, DNS responses were served from Neee-wom's local DNS cache instead of external DNS server. There is a slight increase in the time as the number of hints received increases. Hence, the time taken to process a single hint constitutes a very small portion of the total time taken to query

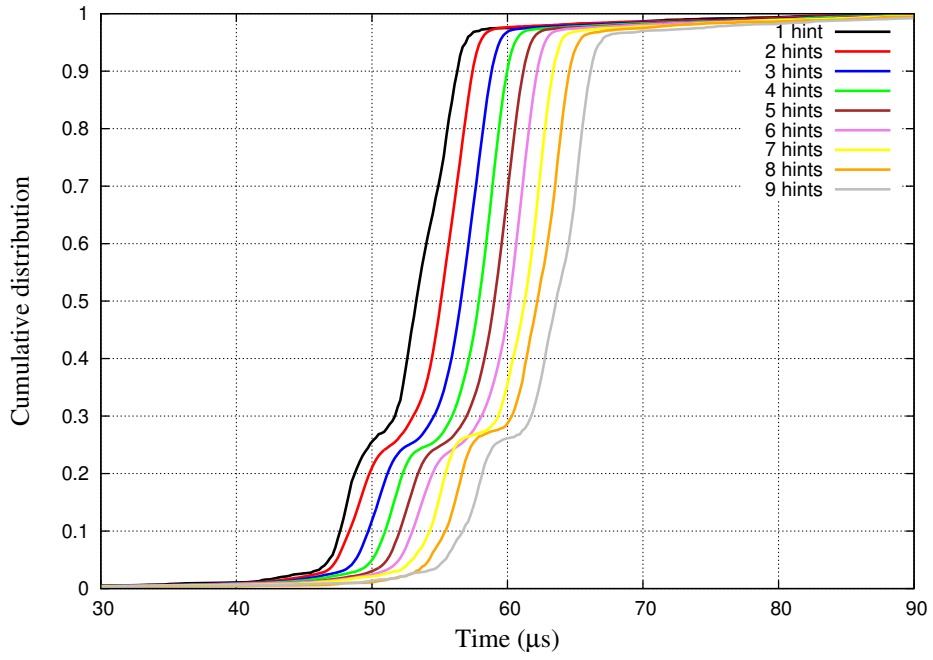


Figure 5.9: Time taken to frame and forward GET request

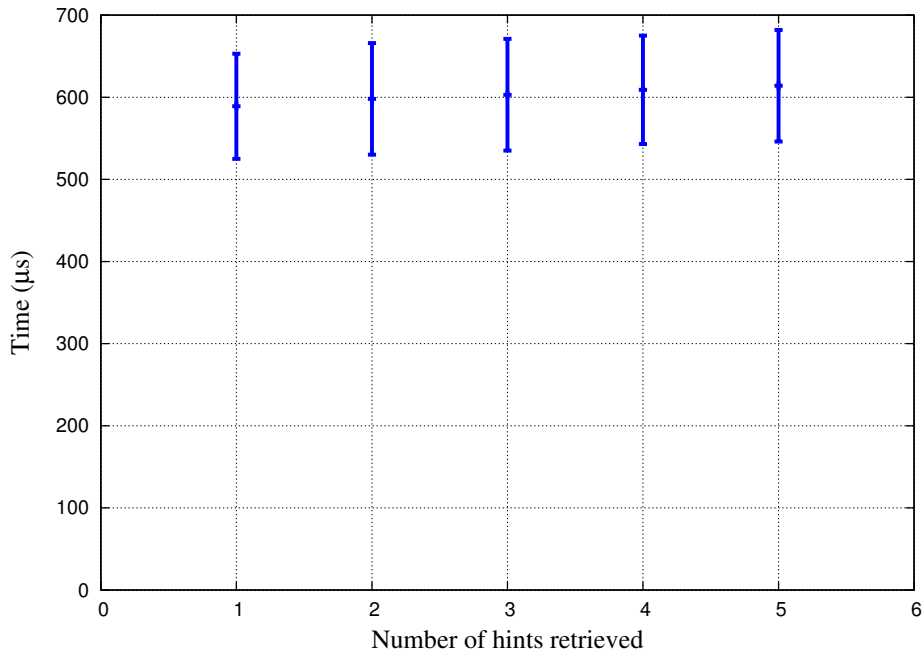


Figure 5.10: Time taken to perform dns hint lookup

DNS. Comparing to the other measurements, the DNS hint lookup time is much larger. The lookup could take even more time if routing hints are served from remote DNS server. Such latency in the routing process is not advisable. However, DNS hint lookup is intended to be performed only at edge nodes. Routing hints received from DNS server are stored in local hint lookup table. Hence, the following GET requests with the same authority part can get routing hints from local table itself.

5.3 Observation

A number of notable observations have been made from the NetInf global routing implementation and the performance measurements. These observations correspond to the global routing solution and the implementation.

NetInf global routing was implemented with the NetInf HTTP convergence layer approach. However, NetInf supports numerous other convergence layer approaches. NetInf messages pass through underlying hop-by-hop transport mechanism. In case of NetInf global routing implementation, a new HTTP connection is opened for each message and closed after each message. Hence, each NNRP node creates new TCP connection to the nexthop node while forwarding each GET request.

As stated before, the nexthop connection establishment takes a the highest time among all the steps. Since Nexthop connection establishment can be optimized by reusing TCP connections, multiple messages can be forwarded by using the same HTTP connection. In addition, each NetInf node waits until the corresponding ACK packets are received. Hop-by-hop transport mechanisms have a significant impact on the overall NetInf message forwarding. Hop-by-hop transport mechanisms that are suitable for NetInf approach in terms of performance and routing scalability should be evaluated. If required, a new hop-by-hop transport mechanism can be defined. So, the scalability and performance of NetInf message forwarding can be improved.

Overall, performance measurements of the newly introduced NetInf global routing steps are analysed in detail. The routing hints extraction and insertion can be improved by using a better message format. The hint lookup and the forward lookup can be optimized further by using more efficient lookup mechanisms. Given the fact that NNRP is a research prototype implementation, performance of a production quality NetInf global routing implementation can be more optimal.

Chapter 6

Conclusion and Future work

A scalable routing solution for global ICN has been implemented in this thesis. A NetInf testbed was built on top of the Internet using the global routing implementation. The performance measurements were collected from the NetInf testbed. The experiment results have been analysed in terms of routing performance and scalability. This chapter presents the conclusions of the thesis and proposes some issues for future work.

6.1 Conclusion

A huge majority of current internet traffic is information dissemination. The current host-centric network architecture is not originally designed for information dissemination. Whereas, Information-Centric Networking (ICN) is a future networking architecture designed to serve global level information dissemination. In ICN, the primary abstraction is named data objects (NDOs), and communication is performed by requesting or providing NDOs. Routers forward a request for an NDO towards a suitable copy of the requested NDO. However, the number of NDOs in the current internet is hugely larger than the number of hosts. Hence, routing scalability in the global ICN is an interesting research problem.

In this thesis, a global routing solution for NetInf has been implemented. The routing solution is called as the NetInf global routing solution that provides a scalable mechanism to forward NDO requests in the global ICN. The routing solution consists of two steps namely hint lookup and forward lookup. Hint lookup step maps the authority part of NDO name to routing hints. Routing hints are mapped to nexthop address in the forward lookup step. NetInf global routing was implemented using NEC NetInf router platform (NNRP). Hint lookup and forward lookup modules were added to NNRP. An experiment testbed was built over the Internet using the NNRP routing implementation. Performance measurements were taken from the testbed experiments. The first and the second research questions are answered by the global routing implementation and the performance evaluation respectively.

The first research question is intended to identify the challenges in implementing NetInf global routing. Though routing hints and the priorities were easily included as part of GET request, the resulting performance impact can be high. The whole NetInf message uses ASCII format. Furthermore, the extension field in the message structure is a JSON encoded string as per the NetInf protocol definition. Hence, the routing hints and the priorities are inserted as the JSON encoded string. The routing hints and the priorities are parsed and extracted from the JSON encoded string. The extracted values are converted to binary format in order to be used in hint lookup and forward lookup. Finally, the values are converted back to ASCII format and added to the forwarded GET request. The routing hints and priorities can be efficiently sent in binary format. The routers use the routing hints and the priorities in binary format when performing routing lookups. The routers do not have to convert the routing hints and the priorities from the JSON encoded string to binary format and vice versa, while forwarding each GET request with the routing hints. Hence, the NetInf message format can be redefined to support the inclusion of binary objects.

The scaling properties of NetInf global routing provide the answer to the second research question. In ICN, the routing layer is located above hop-by-hop transport in the form of NetInf convergence layers. NetInf global routing was implemented using the HTTP convergence layer approach. A new TCP session is established by NetInf routers to nexthop addresses in order to forward each GET request. The performance measurements show that the session establishment time is larger than the rest of the steps in forwarding of GET requests. Hence, the hop-by-hop transport mechanism has significant impact on GET request forwarding. In addition to session establishment time, router spends some time in waiting for the acknowledgment for forwarded GET requests. It is worthwhile to analyse and select a suitable hop-by-hop transport mechanism for ICN.

The time taken to perform the hint lookup and the forward lookup is another scaling property of the NetInf global routing. Considering the fact that NNRP is a research prototype, the time taken by the hint lookup and the forward lookup steps appear to be reasonable. However, the performance of production quality implementation is expected to be much better. Further more, NetInf global routing provides two levels of aggregation. Multiple NDOs are aggregated by using the same authority part in the NDOs name. In addition, the high priority routing hints are aggregated on the low priority routing hints. Hence, NetInf is intended to reduce the number of routing entries in the ICN core routers.

6.2 Future Work

NetInf global routing can be extended and further performance evaluation can be carried out. Also, NetInf protocol can be redefined based on the observations made in this thesis.

The routing implementation in this thesis used a static routing hints forwarding table. The BGP routing mechanism can be used to dynamically populate the routing hints forwarding table.

The NetInf global routing solution can be deployed on a realtime global network such as planetlab [30]. Hence, routing scalability aspects can be identified from a global scale network.

As per the present NetInf naming scheme, there can be only one authority part in an NDO name. Since the same NDO can be served from numerous authorities, NetInf global routing can be redefined to include multiple authority parts in a single NDO name.

There are some future work that relate to the NetInf protocol itself. As stated in the previous chapter, the underlying hop-by-hop transport mechanism has a huge impact on the NetInf global routing. The NetInf router opens a new connection to the nexthop node for each message and closes the connection after forwarding the message. As a result, the time taken to establish the connection to the nexthop node constitutes a major portion of the overall time taken to forward a GET request. Furthermore, the routers also wait for the acknowledgment messages from the nexthop node. The connection establishment time can be avoided by using the same connection to forward multiple NetInf messages. A hop-by-hop transport mechanism that supports scalable forwarding of GET requests should be analyzed and identified.

The NetInf message structure can be redefined to support inclusion of binary data. The binary data such as the routing hints can be sent over the NetInf messages in binary format itself. Hence, the time taken to extract binary data from JSON encoded string and the time taken to encode binary data into the forwarded message can be avoided.

Bibliography

- [1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. Plass, N. Briggs, and R. Braynard, “Networking named content,” *Commun. ACM*, vol. 55, no. 1, p. 117–124, Jan. 2012.
- [2] B. Ahlgren, M. D’Ambrosio, M. Marchisio, I. Marsh, C. Dannewitz, B. Ohlman, K. Pentikousis, O. Strandberg, R. Rembarz, and V. Vercellone, “Design considerations for a network of information,” in *Proceedings of the 2008 ACM CoNEXT Conference*, ser. CoNEXT ’08. New York, NY, USA: ACM, 2008, p. 66:1–66:6.
- [3] B. Ahlgren, M. D’Ambrosio, C. Dannewitz, A. Eriksson, J. Golić, B. Grönvall, D. Horne, A. Lindgren, O. Mämmelä, M. Marchisio, J. Mäkelä, S. Nechifor, B. Ohlman, K. Pentikousis, S. Randriamasy, T. Rautio, E. Renault, P. Seittenranta, O. Strandberg, B. Tarnauca, V. Vercellone, and D. Zeglache, “Second NetInf architecture description,” 4WARD EU FP7 Project, Deliverable D-6.2 v2.0, Apr. 2010, FP7-ICT-2007-1-216041-4WARD / D-6.2, <http://www.4ward-project.eu/>.
- [4] “Scalable and Adaptive Internet Solutions (SAIL).” [Online]. Available: <http://www.sail-project.eu/>
- [5] A. Elmokashfi, A. Kvalbein, and C. Dovrolis, “On the scalability of BGP: the role of topology growth,” *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 8, pp. 1250–1261, 2010.
- [6] X. Zhao, D. Pacella, and J. Schiller, “Routing scalability: An operator’s view,” *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 8, pp. 1262–1270, 2010.
- [7] 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. New York, NY, USA: ACM, 2011, p. 1–6.
- [8] A. Narayanan and D. Oran, “NDN and IP routing – can it scale?” Presentation at ICN side meeting at 82nd IETF, Nov. 2011. [Online].

Available: <http://trac.tools.ietf.org/group/irtf/trac/raw-attachment/wiki/icnrg/IRTF%20-%20CCN%20And%20IP%20Routing%20-%202.pdf>

- [9] G. Kunzmann, D. Staehle, B. Ahlgren, M. D'Ambrosio, E. Davies, A. E. Eriksson, S. Farrell, B. Grönvall, C. Imbrenda, B. Kauffmann, D. Kutscher, A. Lindgren, I. Marsh, L. Muscariello, B. Ohlman, K.-Å. Persson, P. Pöyhönen, M. Shehada, O. Strandberg, J. Tuononen, and V. Vercellone, "D-3.2 (D.B.2) NetInf content delivery and operations," SAIL EU FP7 Project 257448, Deliverable D-3.2, version 1.1, Dec. 2012, fP7-ICT-2009-5-257448/D-3.2.
- [10] C. R. Kothari, *Research Methodology: Methods and Techniques*. New Age International, Jan. 2008.
- [11] P. Baran, "On distributed communications networks," *IEEE Transactions on Communications Systems*, vol. 12, no. 1, pp. 1–9, 1964.
- [12] C. Dannewitz, D. Kutscher, B. Ohlman, S. Farrell, B. Ahlgren, and H. Karl, "Network of information (NetInf) – an information-centric networking architecture," *Computer Communications*, vol. 36, no. 7, pp. 721–735, Apr. 2013.
- [13] S. Farrell, E. Davies, and D. Kutscher, "The NetInf protocol." [Online]. Available: <http://tools.ietf.org/html/draft-kutscher-icnrg-netinf-01>
- [14] N. Fotiou, P. Nikander, D. Trossen, and G. C. Polyzos, "Developing information networking further: From PSIRP to PURSUIT," in *Broadband Communications, Networks, and Systems*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, I. Tomkos, C. J. Bouras, G. Ellinas, P. Demestichas, and P. Sinha, Eds. Springer Berlin Heidelberg, Jan. 2012, no. 66, pp. 1–13.
- [15] 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. New York, NY, USA: ACM, 2007, p. 181–192.
- [16] G. Garcia, A. Beben, F. Ramon, A. Maeso, I. Psaras, G. Pavlou, N. Wang, J. Sliwinski, S. Spirou, S. Soursos, and E. Hadjioannou, "COMET: content mediator architecture for content-aware networks," in *Future Network Mobile Summit (FutureNetw), 2011*, 2011, pp. 1–8.
- [17] D. Meyer, L. Zhang, and K. Fall, "Report from the IAB Workshop on Routing and Addressing," RFC 4984 (Informational), Internet Engineering Task Force, Sep. 2007.

- [18] H. Ma, B. Helvik, and O. Wittner, “An impact of addressing schemes on routing scalability,” *Journal of Communications and Networks*, vol. 13, no. 6, pp. 602–611, 2011.
- [19] V. Khare, D. Jen, X. Zhao, Y. Liu, D. Massey, L. Wang, B. Zhang, and L. Zhang, “Evolution towards global routing scalability,” *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 8, pp. 1363–1375, 2010.
- [20] T. Li, “Design Goals for Scalable Internet Routing,” RFC 6227 (Informational), Internet Engineering Task Force, May 2011.
- [21] G. Huston, “BGP Routing Table Analysis Reports.” [Online]. Available: <http://bgp.potaroo.net/>
- [22] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, D. K. Smetters, B. Zhang, G. Tsudik, D. Massey, C. Papadopoulos *et al.*, “Named data networking (ndn) project,” *Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC*, 2010.
- [23] M. Bari, S. Chowdhury, R. Ahmed, R. Boutaba, and B. Mathieu, “A survey of naming and routing in information-centric networks,” *IEEE Communications Magazine*, vol. 50, no. 12, pp. 44–53, 2012.
- [24] S. Farrell, D. Kutscher, C. Dannewitz, B. Ohlman, A. Keranen, and P. Hallam-Baker, “Naming Things with Hashes,” RFC 6920 (Proposed Standard), Internet Engineering Task Force, Apr. 2013.
- [25] J. Rajahalme, M. Särelä, K. Visala, and J. Riihijärvi, “On name-based inter-domain routing,” *Computer Networks*, vol. 55, no. 4, pp. 975–986, Mar. 2011.
- [26] M. D’Ambrosio, P. Fasano, M. Ullio, and V. Vercellone, “The global information network architecture,” Telecom Italia, Technical Report, 2012, tTGTDDNI1200009.
- [27] G. Kunzmann, D. Staehle, B. Ahlgren, M. D’Ambrosio, E. Davies, A. E. Eriksson, S. Farrell, B. Grönvall, C. Imbrenda, B. Kauffmann, D. Kutscher, A. Lindgren, I. Marsh, L. Muscariello, B. Ohlman, K.-Å. Persson, P. Pöyhönen, M. Shehada, O. Strandberg, J. Tuononen, and V. Vercellone, “D-3.3 (D.B.3) final NetInf architecture,” SAIL EU FP7 Project 257448, Deliverable D-3.3, version 1.1, May 2013, fP7-ICT-2009-5-257448/D-3.3.
- [28] “Apache license, version 2.0.” [Online]. Available: <http://www.apache.org/licenses/LICENSE-2.0>
- [29] M. M. Michael, “Scalable lock-free dynamic memory allocation,” *SIGPLAN Not.*, vol. 39, no. 6, p. 35–46, Jun. 2004.

- [30] "Plantlab." [Online]. Available: <http://www.planet-lab.org/>