Scalability Measurements in an Information-Centric Network

N. Blefari Melazzi, A. Detti, and M. Pomposini

Department of Electronic Engineering, University of Rome Tor Vergata, Roma, Italy
{blefari,andrea.detti,matteo.pomposini}@uniroma2.it

Abstract. Information Centric Networking (ICN) is a new paradigm in which the network layer provides users with content, instead of providing communication channels between hosts, and is aware of the name (identifier) of the contents. In this chapter, first, we briefly describe the FP7 project CONVERGENCE and its approach to ICN. Second, we discuss the needs on measurements required by ICN. ICN is different in several aspects, with respect to the current networking architecture. The measurement needs in an ICN are virtually endless, as designing an ICN is conceptually equivalent to devising a new Internet. Thus, claiming to address this issue in a single chapter would be pretentious. However, the study on ICN is in its initial stage and we want to focus on some of the most pressing and specific aspects of ICN, namely the scalability of its naming and routing functionality. This study is necessary to assess the feasibility of ICN, before addressing other metrics of interest. Thus, the third and main part of the chapter describes our routing-by-name architecture and reports the results of specific measurements on routing issues. Measurements are performed both by means of simulations and by using OneLab, an open, global research network that supports the development of new network services. Our results show that the proposed architecture, designed to improve the scalability of routing tables, is feasible with current technology.

Keywords: Internet Architecture, Future Internet, Information-Centric Networking, Routing, Caching, Scalability, Measurements, Simulation, Test-Bed, Experimental network, OneLab, PlanetLab.

1 Introduction

Information Centric Networking (ICN) is a concept proposed some time ago under different names [1][2], which is attracting more and more interest, recently (see e.g. the papers [3][4][5][6][7][8][9][10] and the projects [11][12][13][14][15][16][17]). ICN proposes a shift from the traditional host-to-host communication to a content-to-user paradigm, which focuses on the delivery of the desired content to the intended users. The basic functions of an ICN infrastructure are to: i) address contents, adopting an addressing scheme based on names (identifiers), which do not include references to their location; ii) route a user request, which includes a "destination" content-name, toward the "closest" copy of the content with such a name; this copy

could be stored in the original server, in a cache contained in a network node, or even in another user's device; iii) deliver the content back to the requesting host.
In our view, the advantages of an ICN are:

1. efficient content-routing. Even though today's Content Delivery Networks (CDNs) offer efficient mechanisms to route contents, they cannot use network resources in an optimal way, because they operate over-the-top, i.e. without knowledge of the underlying network topology. ICN would let ISPs perform native content routing with improved reliability and scalability of content access. This would be a built-in facility of the network, unlike today's CDNs;
2. in-network caching. Caching enabled today by off-the-shelf HTTP transparent proxies requires performing stateful operations. The burden of a stateful processing makes it very expensive to deploy caches in nodes that handle a large number of user sessions. ICN would significantly improve efficiency, reliability and scalability of caching, especially for video [44];
3. simplified support for peer-to-peer like communications, without the need of overlay dedicated systems. Users could obtain desired contents from other users (or from caching nodes) thanks to content-routing and forward-by-name functionality, as it is done today with specialized applications, which, once again, do not have a full knowledge of the network and involve only a subset of possible users;
4. simplified handling of mobile and multicast communications. As regards handovers, when a user changes point of attachment to the network, she will simply ask the next chunk of the content she is interested in, without the need of storing states; the next chunk could be provided by a different node than the one that it would have been used before the handover. Similar considerations apply for multicasting. Several users can request the same content and the network will provide the service, without the need of overlay mechanisms;
5. content-oriented security model. Securing the content itself, instead of securing the communications channels, allows for a stronger, more flexible and customizable protection of content and of user privacy. In today's network contents are protected by securing the channel (connection-based security) or the applications (application-based security). ICN would protect information at the source, in a more flexible and robust way than delegating this function to the channel or the applications [4]. In addition, this is a necessary requirement for an ICN: in-network caching requires to embed security information in the content data-unit, because content may arrive from any network or user node and we cannot trust all nodes; thus, end-users must be able to verify the validity of the received data; caching nodes must make the same check, to avoid caching fake contents;
6. content-oriented quality of service differentiation (and possibly pricing); provision of different performance in terms of both transmission and caching. Network operators (especially mobile ones) are already trying to differentiate quality and priority of content, but they are forced to use deep packet inspection technologies. ICN would let operators differentiate the quality perceived by different services without complex, high-layer procedures [6], and off-load their

networks via caching, a very handy functionality, particularly for mobile operators who can differentiate quality and priority of content transferred over the precious radio real estate;

7. content-oriented access control, providing access to specific information items as a function of time, place (e.g. country), or profile of user requesting the item. This functionality also allows implementing: i) digital forgetting, to ensure that content generated at one period in a user's life does not come back to haunt the user later on, ii) and garbage collection, deleting from the network expired information;
8. possibility to create, deliver and consume contents in a modular and personalized way;
9. network awareness of transferred content, allowing network operators to better control information and related revenues flows, favoring competition between operators in the inter-domain market and better balancing the equilibrium of power towards over the top players;
10. support for time/space-decoupled model of communications, simplifying implementations of publish/subscribe service models and allowing "pieces" of network, or sets of devices to operate even when disconnected from the main Internet (e.g. sensors networks, ad-hoc networks, vehicle networks, social gatherings, mobile networks on board vehicles, trains, planes). This last point is maybe the most important one, especially to stimulate early take up of ICN in selected (and possibly isolated) environments.

On the cons side, ICN has some drawbacks and challenges. A first, obvious, con is that it requires changes in the basic network operation. A second con is that it raises scalability concerns: i) the number of different contents and corresponding names is much bigger than the number of host addresses; this has implications on the size of routing tables and on the complexity of lookup functions; ii) in some proposed ICN architectures [3], delivering contents back to requesting users requires maintaining states in network nodes.

In this chapter, first we briefly describe the approach of the FP7 project CONVERGENCE [16] to ICN. Then, we discuss the needs on measurements required by ICN. ICN is significantly different with respect to the current networking architecture, and poses several new requirements to measurements, which have to be performed both in the current network, to understand some of its aspects useful for the design of ICN, and (experimentally) in the new one. The third and main part of the chapter reports the results of specific measurements performed via simulations, and by using OneLab, an open, global research network that supports the development of new network services [18].

2 The CONVERGENCE Project

The CONVERGENCE project [16] has the aim of designing and evaluating an Information and Communication Technology (ICT) system based on a common and self-contained data unit. The ultimate goal of the CONVERGENCE system is to facilitate, enhance and make more efficient the access to and transaction of resources in networked environments. Resources can be media contents, data about services, or digital representation of real-world objects and people. All information required to attain this objective is embedded within the data unit, including signaling, control, and security information, minimizing the need of using external information or states stored outside the data unit itself (e.g., in network nodes). In the CONVERGENCE system, the basic unit of distribution and transaction is called Versatile Digital Item (VDI).

We can describe the CONVERGENCE system, its features and its expected advantages in terms of four high level components, corresponding also to areas of work and research: the VDI, the applications, the middleware and the network (see Fig. 1).

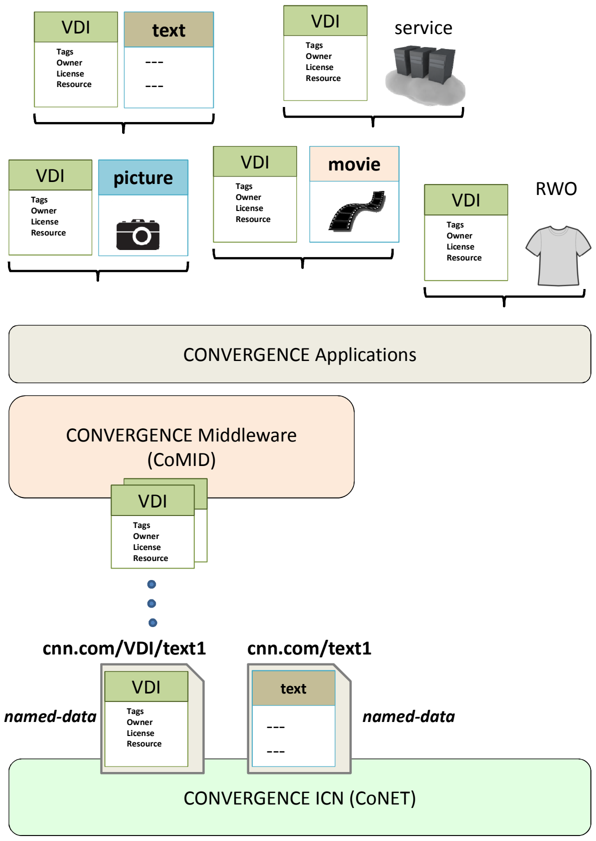


Fig. 1. CONVERGENCE System

2.1 The Unit of Distribution and Transaction

The first area of work is the definition and standardization of a new fundamental unit of distribution and transaction, the VDI. The VDI is a general purpose container which can be used to describe and encapsulate, or make reference to, any kind of resource.

The actual resource can either be physically embedded in the VDI or reside elsewhere and be referenced within the VDI. Resources can be not only classical media files (i.e. texts, pictures and movies), but also data about services, people and Real World Objects (RWOs) (e.g., items of merchandise identified with an RFID). VDIs bind together *meta-information*, which describe the resource, and the *reference to the resource* or the *resource itself* (audio, images, video, text, descriptors of RWOs, descriptors of People, other VDIs, etc.). The meta-data describing the VDI include: i) structural information, describing the content of the VDI; ii) security information (e.g. digital certificates) that enable a recipient to verify integrity and provenance of the resource, and allow legitimate users to decrypt the resource, if necessary; iii) rights information, defining rights to use the resource, and an expiry date for the resource, supporting "digital forgetting".

VDIs are identified by a unique identifier, which is translated (or which is identical) to the network-level name used to route the VDI. The basis for the definition and standardization of the VDI is the MPEG-21 Digital Item [19].

The advantages of having a unique and standard unit of distribution and transaction are easy to understand and include the possibility of defining common mechanisms for handling structured bundles of different and complex information. The availability of these mechanisms will also provide new possibilities for integrating information about RWOs, services and people. Potential beneficiaries include e-auction sites such as E-Bay, location based services, such as "Friend Finder", retail, logistic and goods handling companies. In addition, the combination within the same data unit of data and metadata will allow/simplify several important functionality, as it will be described in the following (e.g. searching functions and web engines operation).

2.2 The Applications

The second area of work is the definition and implementation of tools and of applications, relevant to the needs of business and educational organizations participating in the project, and showing the CONVERGENCE potentiality. Tools are re-usable Application elements, which facilitate re-use of code in Applications; an Application can make use of several tools. Our tools and applications exploit the VDI concept and make use of our middleware and network functionality, so offering to end-users the advantages brought about by our system.

The project designed and implemented four main applications to show the usefulness of CONVERGENCE in four real-life scenarios. The four scenarios are: (i) management of audiovisual material; (ii) management of a large photo archive; (iii) customer relationship management and logistics for the retailing sector; (iv) augmented lecture podcast service enabling a collaborative learning environment.

Other two applications have been built later on by integrating the four main applications; the first integrated application merges the first (video) and fourth (podcasts) original applications; the second integrated application merges the second (pictures) and third (retail) original applications. The aim of the integrated applications is to show that our system is flexible enough to combine different applications in one and to exploit common VDIs.

The advantages of our tools and applications include: i) the provision of basic, easy-to-use functionality to applications developers; ii) the solution of specific needs of consortium partners; iii) the possibility of running real world trials to test the system, and the provision of a basis for future commercial exploitation.

2.3 The Middleware

The third area of work is the definition and standardization of a new open source, extensible, middleware. The CONVERGENCE system supports some sophisticated functionalities (publish/subscribe services, searching functions, security functions), which we think are too complex to be implemented at the network layer, inside routers. Thus, we decided to implement them in a subset of nodes and at the middleware layer.

A first important task of the CONVERGENCE Middleware (CoMid) is the support of a publish/subscribe service model: subscribers register their interest in a resource and are asynchronously notified when publishers make available resources that match their interests. Matching between subscription and publications is based on attributes contained in the VDIs of published resources and on conditions specified in the subscriptions. Publish/subscribe differs from the more traditional request-response service model in a number of ways; the interacting parties do not need to "know" each other. Also, they do not need to know how many subscribers will consume the data they have produced. Publishers and subscribers do not need to interact directly: data consumers will receive the desired data when they will be produced by publishers; publishers do not have to care or check or wait that subscribers consume the data they have produced.

Thus, publish/subscribe effectively decouples the application end-points in space and time. This decoupling of publishers and subscribers offers a much enlarged and flexible typology of services. For these characteristics, publish/subscribe is well suited for disseminating data to a wide and dynamic audience.

The data unit of the CoMid is the VDI.

In practical terms, resource providers using CONVERGENCE will publish VDIs to the middleware, making the middleware aware of the characteristics of the resources of such VDIs. Such awareness enables consumers to subscribe to and receive updates (notifications) both for known resources (e.g. the repair manual for a piece of equipment) or for resources satisfying a given search criteria.

For instance, Alice may be interested in receiving offers for a model of camera she wishes to buy. Alice issues a subscription to the CONVERGENCE middleware describing the camera. When a reseller publishes an offer that matches the request of Alice, she will receive such offers by the middleware. Alice will receive the offers asynchronously, i.e. when connected to the pub-sub system, independently of the connection status of the publisher. Offers are carried by VDIs and, in this scenario, the resource in the VDI could be a web-page where Alice can buy the camera.

It is important to observe that not all CONVERGENCE communications must necessarily use a publish/subscribe paradigm. The CONVERGENCE middleware also accepts direct requests to immediately provide specific requested data, with a traditional request-response service model.

A second important task of the CoMid is to support searches, including semantic searches (see [20] for further details on this CONVERGENCE feature).

A third important task of the CoMid is the provision of security mechanisms for: i) assurance of VDI integrity and provenance (i.e., authenticity of the source); ii) governance of VDI access restrictions and confidentiality; iii) issuing and enforcement of licenses; iv) protection of user privacy.

Our CoMid implements the tasks listed above, providing the following overall advantages:

- Dynamicity of VDIs. The information exchanged between providers and consumers is increasingly volatile. Our CoMid allows producers of information to update the information they have released and consumers to check if a digital resource is up to date, to request an update, and to select between several versions of the same item.
- Privacy and security information *built into the VDI*. This feature avoids the need to delegate privacy and security to applications or to transfer protocols, and ensures that VDIs are genuinely trustworthy. Protecting information at the source is more flexible and robust than delegating this function to applications, or securing only the communications channels.
- Support for "digital forgetting". Our CoMid provides mechanisms allowing users to "unpublish" VDIs and/or to define expiry dates for specific items of information. This ensures that content generated at one period in a user's life does not come back to haunt the user late. Such mechanisms allow sites and services to perform automatic garbage collection, deleting expired information.
- Incorporation of multimedia standards and Semantic Web technologies in VDIs provides a *homogeneous way of searching and handling structured information*.
- CoMid provides interfaces to manipulate VDIs, together with standard mechanisms for producing, managing and linking VDIs with the corresponding metadata. Characteristic examples of these mechanisms include content protection, rights management and event reporting. This facilitates the production and distribution of content in a uniform, interoperable way, compliant to MPEG-M [21] and MPEG 21 [19] standards.
- CoMid provides users with a global identifier for their work (the VDI identifier).

2.4 The Network: Information-Centric Networking

The fourth area of work is the definition and standardization of a new networking functionality. The middleware, implemented in a subset of all network nodes, needs to transfer data (i.e., VDIs) for its own purposes and at the service of applications. Furthermore, applications need to fetch digital resources described by the VDI.

This functionality could be provided by means of standard TCP/IP means.

Instead, CONVERGENCE has taken an alternative approach, which is more consistent with the use of a common and self-contained data unit at the application and middleware level.

The chosen approach is Information Centric Networking (ICN), briefly described in the Introduction. Our CoNet defines its own data unit at the network layer, called *CONET Information Units (CIUs)*: *interest CIUs* convey requests of named-data (e.g. a VDI); *named-data CIUs* transport chunks of named-data. Named-data is any digital object, uniquely identified by the network with a name (i.e. a string). A named-data can be a VDI or the actual resource referenced to by the VDI. For instance, in Fig. 1 we have both cases: the VDI of text1 of cnn.com, and the actual text1 file of cnn.com. At the network-level, both are named-data; the former is identified by the string “cnn.com/VDI/text1”, the latter by the string “cnn.com/ text1”.

We identified eight fundamental issues that need to be addressed to design an ICN infrastructure:

1. Primitives & interfaces, which define the relationship of the ICN protocols with the overall architecture.
2. The naming scheme, which specifies the identifiers for the data units (CIUs) addressed by the ICN.
3. The route-by-name mechanism, used by ICN nodes to relay an incoming CIU to an output interface. The output interface is chosen by looking up a “name-based” forwarding table.
4. The routing protocols used to disseminate information about location of CIUs, so as to properly setup the name-based forwarding tables.
5. The data forwarding mechanism that allows CIUS to be sent back to the device that issued a CIU request. Data forwarding cannot use the forward-by-name mechanisms, because, typically, devices/interfaces are not addressed by the content routing plane of an ICN.
6. In-network caching, which concerns the ability of ICN nodes to cache CIUs and to reply to incoming CIUs requests.
7. Segmentation & transport mechanisms (see e.g. [9]) needed to: 1) split a whole content (e.g. a VDI) in different CIUs (or chunks); each CIU is an autonomous data unit with embedded security and addressable by the routing plane; ii) ensure a reliable transfer of CIUs from the origin node (or from a cache node) towards the requesting node; iii) counteract congestion.
8. Security & privacy issues tackling (at least) three specific aspects: 1) how to guarantee content authenticity and protect the network from fake content, which could also pollute network caches; 2) how to guarantee that content be accessed only by intended end users, and 3) how to protect information consumers from profiling or censorship of their requests.

Finally, the network should complement mechanisms provided by the Middleware for the support of the “digital forgetting” and garbage collection functionality. For instance, the network should not forward content whose expiry date is terminated.

The Convergence Network (CoNet) is designed according to these principles.

The advantages of ICN in general and of our CoNet in particular are described in the Introduction.

3 Measurements Needs in an ICN

The measurement needs in an ICN are virtually endless, as designing an ICN is equivalent to devising a new Internet. Thus, claiming to address this issue in a single chapter would be pretentious. However, the study on ICN is in its initial stage and we want to focus on some of the most pressing and specific performance aspects of ICN, namely the scalability of its naming and routing functionality. This study is necessary to assess the feasibility of ICN, before addressing other metrics of interest.

Once the theoretical feasibility of ICN is demonstrated, one could go and study the performance of the other fundamental functionalities, which we listed in the previous section, and to assess the advantages of ICN, as identified in the introduction.

Thus, in this chapter, we focus on measurement issues regarding the scalability of routing-by-name functions, assuming that the ICN is used to fetch current Web contents.

4 Routing-by-Name

In this Section, we briefly recall our reference model [7], and our Lookup-and-Cache solution [10][22], which implements the routing-by-name functionality.

4.1 Reference Model

ICN nodes (see Fig. 2) are interconnected by "sub-systems" [7]. Sub-systems use an underlying technology to connect ICN nodes and can be implemented in several different ways. For instance, a sub-system could be a public or private IP network, an overlay UDP/IP link, a layer-2 network, a PPP link, etc. This is the same concept used in current IP networks, in which different layer 2 technologies connect IP hosts and routers. Nodes can be: ICN end-nodes (or clients) that download contents; ICN serving-nodes (or servers) that provide contents and ICN nodes that relay ICN data-units between sub-systems, which may also cache data.

To provide a content, a server splits the content in blocks of data, named *chunks*, and assigns a unique network identifier to each chunks. A network identifier is a string like "cnn.com/text1.txt/chunk1", which is said to be the "name" of the chunk.

In the CONVERGENCE system, the name could be equal to the VDI identifier or derived from it.

The role of the ICN protocols is to discovery and deliver named chunk. In order to fetch a chunk, a user issues a data unit, named *Interest* message, which contains the name of the chunk. ICN nodes *route-by-name* the Interest message, by using a longest prefix matching forwarding strategy and a name-based routing table. We name the entries of the name-based routing table *ICN routes.* An ICN route has a format like:

<name-prefix, next hop >

A name-prefix should be either the full name of a chunk, e.g. "cnn.com/text1.txt/chunk1", or a continuous part of it, starting from the first left character e.g. "cnn.com/".

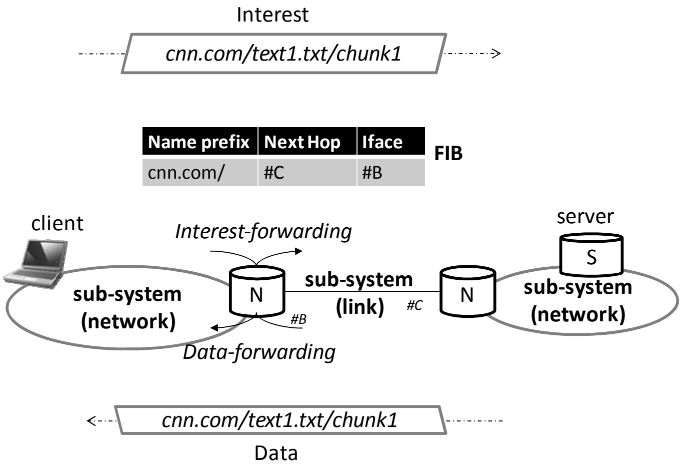


Fig. 2. Network model

The first "en-route" device, be it an intermediate node or the end-server, that has the chunk sends it back within a data unit, named *Data* message, which includes the chunk name. Network nodes forward the Data message towards the requesting client, through the same sequence of ICN nodes previously traversed by the Interest message. These nodes may store the Data in their cache, so as to provide a so-called en-route[1], in-network, caching service. The Data forwarding process exploits reverse-path information either temporary stored in the traversed nodes during the Interest forwarding process (see Pending Interest Table of [3]), or contained in the header of Data message, and previously collected in the Interest message during its forwarding process (see reverse-path source-routing in [7]). Therefore, the routing-by-name process does not involve Data messages, but only Interest messages.

Downloading a whole content is achieved by sending a *flow* of Interest messages to retrieve all the chunks of the content. The sending rate of Interest message is regulated by a receiver-centric congestion control mechanism [23][9], which could be based on the same logic used by TCP. Therefore, in our ICN model, we have endpoints that exchange Interest-Data sequences and the message exchange rate is

[1] We point out that en-route caching does not have an impact on the routing plane. Indeed the routing-plane only routes-by-name requests toward servers. Conversely, in case of off-route caching, the routing-plane should route-by-name requests towards cached contents. The temporal dynamics of these additional "caching routes" is a function of the lifespan of contents in caches, which could be very short. This could cause an excessive routing traffic and processing load. For this reason, an Information Centric Network typically adopts only en-route caching.

regulated by the receiver. Dually, in TCP/IP the endpoints exchange Segment-Ack sequences and the exchange is regulated by the sender.

As regards the naming scheme, several proposals (e.g. [2][3][4][24]) agree in adopting a hierarchical naming. In this chapter, we assume a rather general hierarchical naming scheme where a name is formed by a sequence of *Components*; i.e. a name has the form "Component_1/Component_2/../Component_n". This scheme supports current Web URL, where the Component_1 is the domain name (e.g., "cnn.com") and next Components represent the path of the local resource (e.g., /text1.txt). In addition to these Components, which represent the *content-name*, ICN requires other specific Components, e.g. to represent the chunk number ("/chunk1"), version, etc. The full sequence of Components is referred to as the *chunk-name*.

As said before, in this chapter we focus on a scenario in which the ICN is used to distribute current Web contents and Web servers are replaced by ICN servers. Usually, a Web server provides all contents whose URLs have the same domain-name, e.g. "cnn.com". Therefore, we assume that an ICN sever provides all contents whose names have the same Component_1, which is equal to the domain-name. In this scenario, we argue that the minimal set of routing information needed to route-by-name contents offered by ICN servers depends on the number of domain-names, rather than on the number of content-names or chunk-names. Hence, the name-prefix of an ICN route is a domain-name and, therefore, the number of ICN routes that a node of the default-free-zone should handle is in the order of the current domain-names, i.e. 2 10^8; we assume 10^9 to have some margins [10][22].

We remark that these conclusions are dependent on the assumptions stated above.

Changing the assumption would change the results. For example: i) using a "flat" non-hierarchical naming the number of ICN routes would be higher and likely close to the number of content-names, i.e. 10^{11}; ii) if we allow more than one route per name-prefix, e.g. for routing redundancy or multi-homing purposes, the number of ICN routes would be higher than 10^9; iii) nodes that have a default route, e.g. corresponding to a tier-2 or a tier-3 node of the current Internet, would have a number of ICN routes much lower than 10^9, and so forth.

4.2 Lookup-and-Cache Routing Architecture

The routing-by-name of Interest messages is very similar to the routing of IP packets but, in place of IP-prefixes, the routing-by-name procedure uses name-prefixes, which, in our "fetching web contents" scenario are domain-names. Consequently, it is worth analyzing the feasibility of reusing the architecture of an IP router for an ICN node.

A typical router is composed of three major components: one or two routing engines, line cards that host a forwarding engine and a switch fabric. The routing engine handles the routing protocols and stores the routes in a routing table, named Routing Information Base (RIB). In general, the RIB contains several routes to the same destination and it is implemented by means of cheap and slow memories such as DRAM. The forwarding engine of a line card receives incoming packets and selects the output line card by looking up an on-board routing table, which is named

Forwarding Information Base (FIB) [25][26]. The FIB contains one route per destination, and therefore a smaller number of routes than the RIB. To support packet forwarding at line rate, the forwarding process is carried out by dedicated ASIC chips and the FIB is implemented with fast memories, such as SRAM or TCAM. These memories are expensive, consume a lot of power, and do not follow Moore's Law [27]. After the selection of the output interface, the forwarding engine injects the packet in the switching fabric. The switching fabric is (at least conceptually) an *N*x*N* non-blocking crossbar where *N* is the number of line cards.

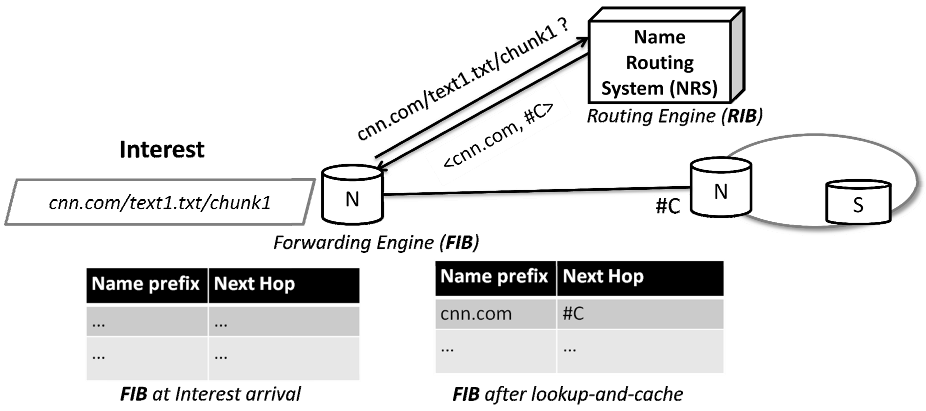
If we want to reuse this architecture to route-by-name ICN Interest messages, we should store ICN routes in the FIB and RIB, and properly update the routing and forwarding logics. Hence, a fundamental check is to verify the practical feasibility of storing all required routes in a FIB and in a RIB. As regards the FIB, the maximum size of a SRAM chip is today 32 MByte [28]. Assuming that an ICN routing entry is 45 bytes long [2], the number of routing entries storable in a FIB is in the order of 10^6 (i.e. 32MB/45B). In the previous section we estimated that an ICN node should handle 10^9 routes and thus current FIB technology cannot store the whole set of ICN routes.

Let us now analyze the RIB issue. As in IP, the RIB would contain more than one route per name-prefix; this redundancy mainly depending on the peering relationships among Autonomous Systems. For instance, current BGP data obtained from the AS6447 node [29] show that, on average, its RIB contains 31 routes per destination. As a consequence, we assume that the RIB of an ICN node should handle a number of routes in the order to 10^{10}, i.e. one order of magnitude greater than the number of name-prefixes. In this case, the RIB would require hundreds of Gbytes (i.e., 10^{10} *45B) of DRAM memory and a motherboards with hundreds of memory slots. Current DRAM chips are of 4 GB and motherboards of “expensive” carrier-grade IP routers can host up to 4 memory slot [30][31]. This means that the required increase of capacity is in the order of 10^2. We can conclude that supplying each network node with a motherboard with hundreds of memory slots would dramatically increase the deployment cost of an ICN network, with respect to an IP network.

In order to cope with the capacity issue of the FIB and with the cost issue of the RIB, we propose a *Lookup-and-Cache* routing architecture. In our solution, we use the FIB of a Forwarding Engine as a *route cache* and deploy a *centralized routing engine*, that runs on a server named Name Routing System (NRS), which logically serves all the ICN nodes of a sub-system. Fig. 3 reports an example of Lookup-and-Cache operations. Node *N* receives an Interest message for “ccn.com/text1.txt/chunk1”. Since the FIB lacks the related route, the node temporarily queues the Interest message, lookups the route in a remote RIB, gets the routing information and stores it in the FIB, and then it can forward the Interest message. In what follows, we discuss the rationale underlying the Lookup-and-Cache architecture.

4.2.1 FIB as a Route Cache

It is well-known that the relative frequency with which Web contents are requested follows the Zipf’s law [32] and that there is a time and space locality of Web content interests. Therefore, a large number of *flows* of Interest messages that an ICN node should *concurrently* route-by-name refer to a small set of contents and, more important, these flows use an even much smaller set of ICN routes, since ICN routes address domain-names rather than single contents. In Section 5, we show that the set of these *active-routes* can be comfortably stored in a SRAM memory. Therefore, we propose to use the FIB as a route cache, which should contain, at least, the entire set of active-routes. When the FIB lacks a route, the node lookups the route in a "remote" RIB and then caches the route in its FIB. When all FIB rows are filled in, new routing entries may substitute old ones, according to a specific *route replacement algorithm.* Furthermore, a routing entry could be removed or updated by a *FIB-RIB consistency mechanism* [22].



Name prefix	Next Hop
...	...
...	...

Name prefix	Next Hop
cnn.com	#C
...	...

Fig. 3. Lookup-and-Cache concept

4.2.2 Centralized Routing Engine

All ICN routes are contained in the RIB of a Routing Engine, which logically serves all the ICN nodes of a sub-system and runs on a centralized server, named *Name Routing System* (NRS) node. Thus, the cost of an expensive Routing Engine is taken for only one network device, rather than for all network nodes. Of course, the DRAM memory of the NRS node must be able to contain all the ICN routes of all the ICN nodes that it serves. A single NRS node may also serve more than one sub-system; for instance all sub-systems administered by the same company (e.g. a whole autonomous system).

Since many Interest flows use a small set of active-routes, the temporal dynamics of *active*-routes is slower than the flow dynamics. Indeed, a route is used for a period of time that is greater or equal than/to the duration of a single flow. This limits the lookup rate that a centralized Routing Engine should deal with and, in Section 5, we show that this rate is easily supported by current technologies.

So far we have described the "data-plane" of our Lookup-and-Cache architecture, i.e. the procedures carried out to forward ICN messages. In addition to the data-plane, the Lookup-and-Cache architecture (as the IP one) needs "routing-plane" procedures that run on NRS nodes and whose goal is to setup the RIBs. The routing-plane is out of the scope of this chapter; anyhow we point out that our architecture does not impose a specific routing-protocol. For instance, we can support both name-based version of BGP, as suggested in [3], or DONA [2], where the DONA Resolution Handler (RH) has the same function of the NRS node. We conclude by observing that, as it occurs in the current Internet for BGP messages, the ICN nodes should give highest priority to routing signaling messages (e.g., lookup and routing messages), to limit the number of failed communication attempts and the delay.

4.2.3 Route Replacement Algorithm

When a node receives an Interest message for a given content and it is not possible to find a matching route in the FIB, we have a *route-cache-miss* event. In this case: i) if the FIB is not full, the node performs a lookup in the remote RIB and stores the new route in the FIB; ii) the forwarding of the Interest messages is subject to a route-lookup delay. When the FIB is full, the insertion of a new route implies the replacement of an old route. In this case, a *route replacement algorithm* decides whether to lookup the new route or not. In the first case it also decides which old route has to be replaced. In the second case, the Interest message is dropped and subsequently retransmitted by transport level mechanisms.

An inefficient design of the route replacement algorithm would result in an excessive rate of route lookups, with a consequent worsening of delay performance (as more Interest messages will be subject to the route-lookup delay) and an increase of the load of the NRS node. To mitigate these inconveniences, it would be desirable to replace *inactive* routes. Consequently, the design of the route replacement algorithm aims at solving two problems: first, how to identify *inactive* routes and, second, how to behave in case of *FIB overload*, i.e. when there are no inactive routes and a new route needs to be added in the FIB.

In [10][22] we proposed a route replacement algorithm, which assumes that each route contained in the FIB has an inactivity time out (ITO), after which the route is considered inactive; its performance are compared with the Least Recently Used (LRU) policy [43]. Results show that, if the FIB size is over dimensioned and the FIB operates in an unloaded condition, the least recently used route is likely inactive; hence the simple LRU works well, as the more complex ITO. If the FIB size is under dimensioned and the FIB works in overload condition, ITO overcomes LRU as LRU causes an in/out flapping of routes from the FIB.

5 Feasibility Check

In this section we show that our architecture is feasible by using currently available technology. To this end we verify that: i) the capacity provided by current FIB technology is enough to store the expected number of active-routes; ii) the route lookup rate can be supported by current database technology.

On a given node and at a given time, an ICN route is "active" if there is at least one flow of Interest messages using that route. This concept is sketched in Fig. 4, where there are 3 flows of Interest messages toward "cnn.com". The route toward "cnn.com" becomes active at the start of the first flow and becomes inactive at the end of the last flow. In Fig. 4 there is also a single flow of Interests for "bbc.com", thus the related route activity has the same duration of the flow.

In the current Internet, a client sends TCP ACK and receives TCP segments from the Web server. In an ICN, a client sends Interest messages and receives Data messages from the ICN server, or from an en-route cache. So, if a client used the ICN to download Web contents, then the traditional flows of TCP ACK messages would be replaced by a flow of Interest messages. Furthermore, on the base of our hierarchical naming assumption, the couple <IP destination address, destination Port> contained in TCP ACK messages would be replaced, in Interest messages, by a chunk-name that contains the domain-name of the destination Web server.

For instance, assume that in the current Internet a host sends an HTTP request towards the domain name "cnn.com". The domain name "cnn.com" will be translated by DNS into an IP address, e.g. 157.166.226.25, a request will be sent to this address and then the data will be directed from 157.166.226.25:80 towards the requesting host, while a flow of TCP ACKs will be directed by the client to 157.166.226.25:80. In the proposed ICN scenario the flow of TCP ACKs would be replaced by a flow of Interest messages for chunks, whose names contain the "cnn.com" name-prefix.

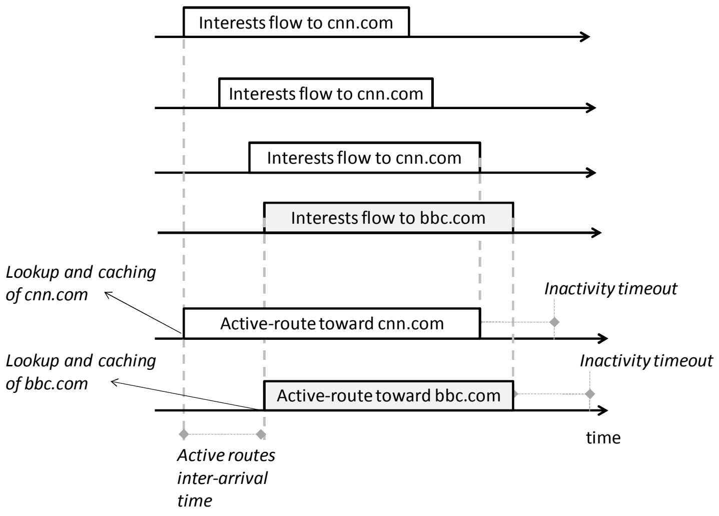

Interests flow to cnn.com
Interests flow to cnn.com
Interests flow to cnn.com
Interests flow to bbc.com
Lookup and caching of cnn.com
Inactivity timeout
Active-route toward cnn.com
Lookup and caching of bbc.com
Inactivity timeout
Active-route toward bbc.com
time
Active routes inter-arrival time

Fig. 4. Flows and active-routes

Using such a mapping between the flow of TCP ACKs and the flow of ICN Interests, we could use current Internet traces to assess the feasibility of ICN. We could replace each ACK of an Internet trace with an Interest message, thus creating a would-be ICN trace. Unfortunately, IP traces usually have anonymized IP addresses, which do not include the domain-names of HTTP GET messages. Hence, we cannot derive the domain-names to be used for the conversion from TCP ACKs to Interest messages, by using such anonymized traces.

To circumvent this problem, we use a simulation approach to associate a domain-name to a flow of TCP ACKs directed towards an anonymized IP address.

The simulation model is depicted in Fig. 5. Briefly, for a given anonymous trace, we randomly associate the web servers' anonymous IP addresses of the trace to a set of public IP addresses, derived (as described below) from the 1 million most used domain-names [33]. Then, we associate each anonymous flow of the trace to a domain-name, randomly extracted among those domain-names that have the public IP address associated with the web server's anonymous IP address of the flow.

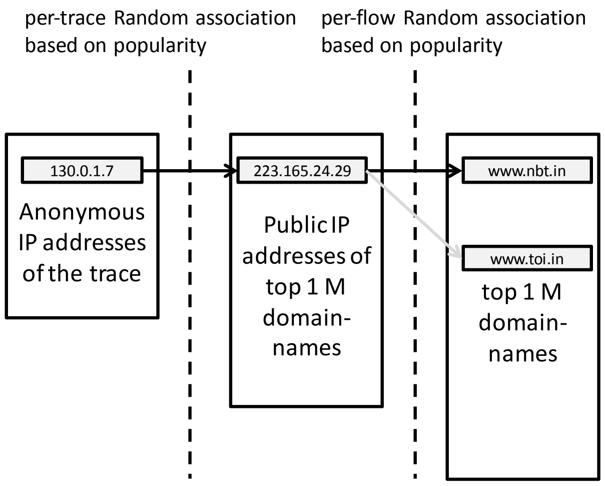

per-trace Random association based on popularity
per-flow Random association based on popularity
130.0.1.7
Anonymous IP addresses of the trace
223.165.24.29
Public IP addresses of top 1 M domain-names
www.nbt.in
www.toi.in
top 1 M domain-names

Fig. 5. Simulation model to associate an anonymous IP address to an actual domain-name

More in details, the simulation model is formed by three phases, as follows:
<u>Phase-1: data structures setup</u>

1. we collect the top 1 million domain-names in a list named {*DN*};
2. for each domain-name DN_i, we model its occurrence probability $opDN_i$ in an Internet trace as a function of its rank position, and according to a Zipf's law. Following the results of [34] we set the value of the Zipf alpha parameter to 1[2];
3. we resolve the list {*PubIP*} of public IP addresses associated to each domain-name[3] (from a machine located in the campus of University of Rome Tor Vergata);
4. for each element $PubIP_i$, we compute its occurrence probability $opPubIP_i$ as the sum of the occurrence probability $opDN_j$ of the domain-names that use the IP address $PubIP_i$;
5. from the anonymized trace, we extract the list {*AnIP*} of unique anonymized IP addresses of web servers;

[2] We remind that we are considering the occurrence distribution of domain-names, rather than that of specific contents, and that the parameter alpha of the domain-name Zipf [34] is greater than the one of the content Zipf [32] (e.g, 0.6, 0.8).

[3] Since the same IP address may serve several domain-names, the number of unique elements of {*PubIP*} is lower than the length of {*DN*}. In our case the ratio between the length of {*DN*} and the number of unique elements of {*PubIP*} is equal to about 1.7.

6. for each element $AnIP_i$, we compute its occurrence probability $opAnIP_i$ as the ratio between the number of HTTP flows that have $AnIP_i$ as destination address and the total number of HTTP flows of the trace.

Phase-2: Random association of anonymous IP addresses to public IP addresses

7. since the number of public IP addresses $\{PubIP\}$ is in the order of 580k while the number of anonymous IP addresses of our trace is lower, we randomly extract a subset of public IP addresses, by using their occurrence probability $\{opPubIP\}$. We refer to this restricted set as $\{rPubIP\}$;
8. we map, one-to-one, elements of $\{AnIP\}$ to elements of $\{rPubIP\}$. We preventively sorted the elements of $\{AnIP\}$ and of $\{rPubIP\}$ on the base of the occurrence probabilities of their elements. Consequently, the element of *AnIPk* with rank k in terms of occurrence probability is mapped to the element *rPubIPk* that has the same rank.

Phase-3: Random association of anonymous flows to domain-names

9. for each flow of the trace, we map its destination anonymous IP address $AnIP_i$, to the public address $rPubIP_i$ and we randomly associate to it a domain-name randomly extracted among the ones that use $rPubIP_i$. The extraction is properly weighted by the occurrence probability $opDN_i$.

Since each flow has now an associated domain-name, we can convert the TCP ACKs of a flow in Interest messages, and evaluate the average number of ICN active-routes and the average active-route inter-arrival time by using real Internet trace. Results are reported in Table 1. The Equinix-sanjose-* and Equinix-chicago-* traces [35] are captured on a 10 GigE interfaces of a tier-1 ISP. The Mawi-* traces [36] are captured on a trans-Pacific line operating at 150 Mbit/sec. The Rome-Tor-Vergata trace is captured on the 1 GigE interface of the router gateway of our University [37], which is a tier-3 network. Even in the worst case of the Equinix-sanjose-dirA trace, the average number of active-routes is in the order of 10^3; this value is much lower (by a factor of 10^3) than the capacity provided by an off-the-shelf SRAM based FIB, i.e. 10^6 ICN routing entries, as discussed in Section 4.2.

Table 1. Average number of active routes and inter-arrival times

Trace id	Average value of ICN active-routes (N_{icn})	Average ICN active-routes inter-arrival (I_{icn})
Equinix-sanjose-dirA	4680	0.5 ms
Equinix-sanjose-dirB	1782	1.1 ms
Equinix-chicago-dirB	1576	1.2 ms
Mawi-1	250	4.5 ms
Mawi-2	267	3.3 ms
Rome-Tor-Vergata	185	5.6 ms

Let us now investigate if current database technology can support the required lookup rate. Table 1 reports that the average inter-arrival time between the starts of two consecutive active-routes is in the order of half a millisecond, for the worst trace.

When the FIB memory is dimensioned for containing all active-routes, the inverse of the active-routes inter-arrival time is an upper bound of the lookup rate. Indeed, we need a lookup at the start of the route activity only if that route is not already cached in the FIB. Therefore, an average active-route inter-arrival time in the order of 0.5 ms implies a lookup rate in the order of 2000 lookups per second, in the worst case. This value is easily achievable with current database technology. For instance, we have implemented an NRS node with a Bind9 server, running on an old Linux laptop with an Intel Pentium Processor M at 1.4 Ghz, and we measured a sustainable rate of about 15 000 lookups per second.

We also evaluated the number of active-routes versus time for the Equinix-sanjose-dirA trace (Fig. 6). The number of active-routes has a limited variation around its average value. This simplifies the dimensioning of the FIB size, which can be set close to the observed mean, without requiring a large margin.

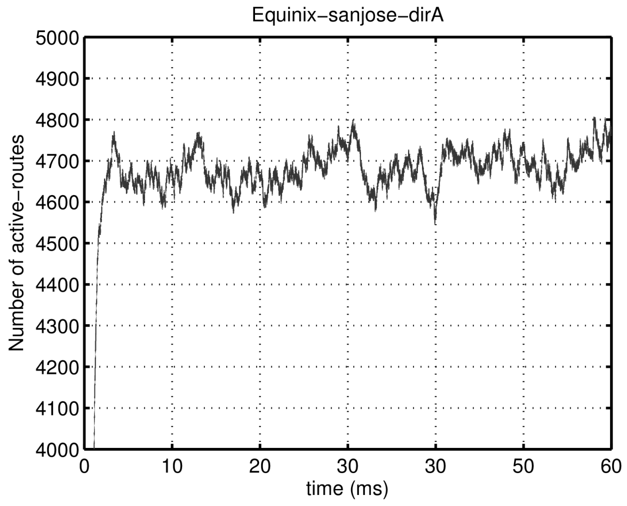


Fig. 6. Number of active-routes for the Equinix-sanjose-dirA trace

Finally, we investigated the effectiveness of FIB *over-provisioning*, to reduce the lookup rate. A FIB is said to be over-provisioned, when it has a capacity significantly greater than then average number of expected active-routes. For this analysis we used an ideal route replacement policy that randomly replaces inactive-routes. Fig. 7 shows the resulting lookup rate vs. the FIB size for the Equinix-sanjose-dirA trace; we observe a significant reduction of the lookup rate as the FIB size increases.

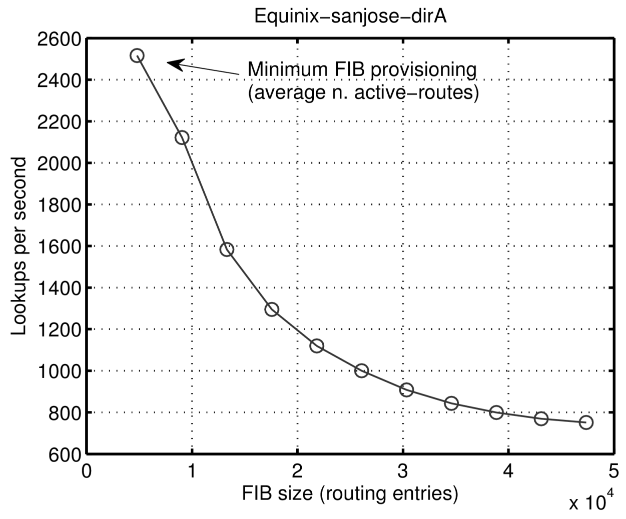


Fig. 7. Lookup rate versus FIB size for the Equinix-sanjose-dirA trace

6 Experiments on OneLab

In this section we show the functionality of Lookup and Cache architecture and evaluate its main performance by using the OneLab test-bed facility [18]. Specifically, we use 20 devices, located in different countries and belonging to the PlanetLab Europe network [18]. We implemented our Lookup-and-Cache architecture with a software package, mainly composed of a modified version of CCNx 0.5.0 [38] and a Java-based implementation of the NRS node. All the software is available in [39]. For the FIB replacement algorithm, we used LRU [43].

We analyzed the case of an ICN network formed by 19 ICN nodes and by a single centralized NRS node. The network topology is shown in Fig. 8, where each ICN node is marked with the country code of the supporting PlanetLab device. The NRS is located in Ireland. The connectivity graph of the network resembles a subset of the Pan-European GEANT research network [40].

As shown in the figure for the IE node, we assume that each ICN node serves a subsystem, containing ICN clients and ICN servers. Furthermore, each ICN node is connected with its neighbors by means of an overlay UDP/IP link. We setup this overlay network by properly configuring the next hop of the ICN routing tables. For instance, the ICN routing table of the IE node has the UK node as next-hop for any content, with the exclusion of contents published by the ICN server handled by the IE device.

This scenario may represent, for instance, the case of a single Autonomous System that uses ICN technology to exchange contents located in internal servers and, to this aim, deploys 19 ICN nodes/sub-systems, whose routing-by-name function is controlled by a centralized NRS device. To simplify the test-bed, we virtualize all the ICN servers and ICN clients contained in an ICN sub-system by using only one client and only one server, both contained in the PlanetLab device that hosts the ICN node of the sub-system. Therefore, each PlanetLab device of Fig. 8 (excluding the NRS) has the role of ICN client, ICN server and ICN node.

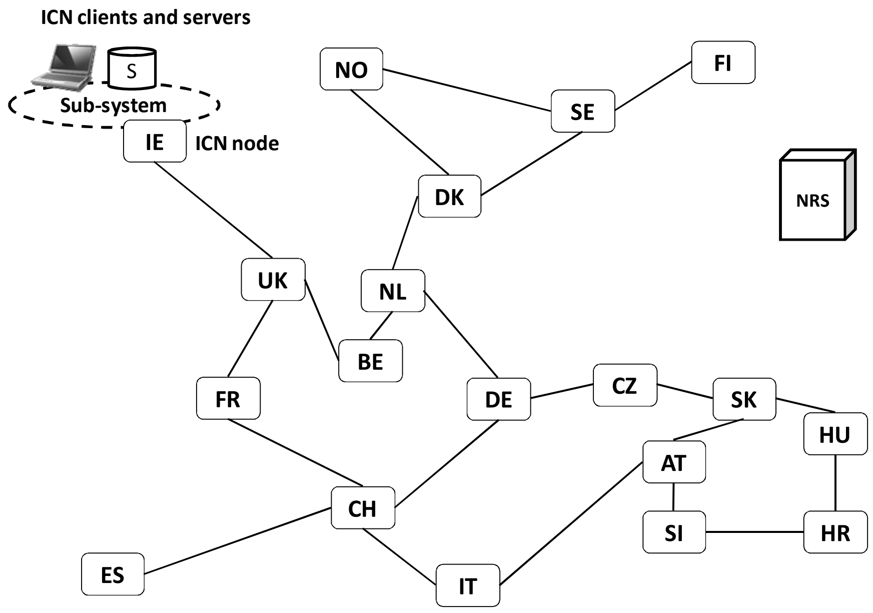


Fig. 8. ICN topology implemented on PlanetLab

Each ICN server handles 20 unique domain-names and, for each domain name, it publishes 5 contents of 500kB. For instance, a server that handles the domain-name www.cnn.com, publishes the contents www.cnn.com/text1.txt, www.cnn.com/text2.txt, ... , www.cnn.com/text5.txt. Therefore, in the whole network we have 380 domain-names, i.e. ICN routes, and 1900 contents (231800 chunks), uniformly distributed among network nodes. Each client generates 300 requests of contents with an inter-arrival time that follows a negative exponential distribution, with average 4s. To select a content, a client first chooses the domain-name according to a Zipf distribution with alpha=1 [34], then it randomly singles out one of the 5 contents associated to the selected domain-name.

The NRS node contains the ICN routes of all 380 domain-names, for all 19 ICN nodes. To compute the ICN routes, the NRS node uses a shortest path routing on the topology depicted in Fig. 8. For instance, a content request issued by the ES node for a content stored in the UK node is routed-by-name on the path: ES-CH-FR-UK.

ICN nodes do not use default routes, even though this could be possible in case of leaf nodes, for instance the IE one. Therefore, each time that a node has to forward an Interest message, if it does not have the related route in its FIB, it has to lookup-and-cache that route, by querying the NRS node.

The queries to the NRS node use a direct UDP/IP connection (not reported in the figure) between the ICN node and the NRS node. Therefore, the signaling traffic between ICN nodes and NRS is not routed-by-name on the ICN topology of Fig. 8, but it is transferred by using underlying, traditional, IP means. As a future work, we will support also NRS queries with ICN means, as proposed in [10].

To conclude the description of the test-bed, we remind that, in addition to the FIB memory that we use as a *cache of routes*, an ICN node has a storage space used as a *cache of content-chunks* (i.e., a cache of network layer data units, CIUs) to implement the in-network caching functionality discussed in the introduction (second advantage, in-network caching). In our ICN nodes, we use the default CCNx content cache replacement algorithm, i.e. FIFO.

Fig. 9 shows the average download time versus the FIB size, comparing the case of nodes without content cache and the case of nodes with a content cache; the content cache size is equal to 10% of the total number of published chunks. The x-axis includes also an out-of-scale point, representative of a full preloaded FIB (labeled "Full-FIB") where, for each node, we use an *unlimited* FIB, pre-loaded with all ICN routes that the node could use. This measurement allows highlighting the worsening of performance deriving from the use of a *limited* FIB as a cache of routes and from the use of a centralized remote RIB.

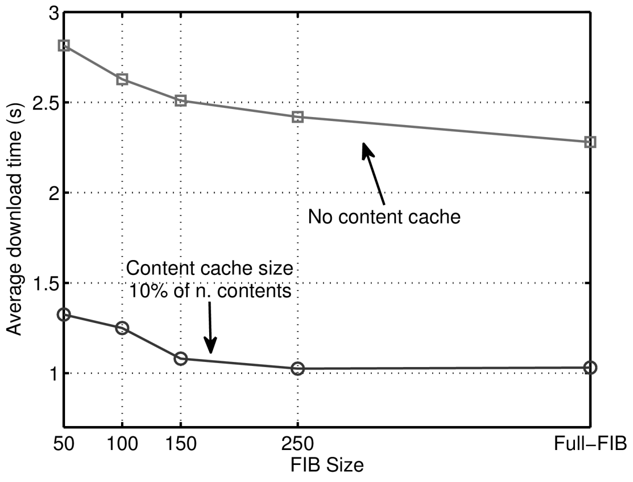


Fig. 9. Average download time versus FIB size

As expected, as the FIB size increase, the performance tends to the full-FIB case, while caching contents leads to a decrease of the download time as some chunks are delivered by the cache of nearby nodes, rather than from far away servers.

If we look at the curve representing the no content cache case, the download time decreases of about 600 ms, when the FIB size increases from 50 to the full-FIB case. We argue that this delay is due to the connectivity/processing delay brought about by the NRS node. This lookup delay (in the worst case equal to about 350 ms) would not occur if the traffic from/to the NRS had priority on the other user traffic and if the NRS were implemented by using a suitable powerful hardware.

Fig. 10 shows the number of lookups per second, measured at the NRS node, versus the FIB size. As measured in real Internet traces (see Fig. 7), also in the OneLab test-bed we obtain a significant reduction of the lookup rate by increasing the FIB size.

However, we have been surprised to see that the in-network caching of contents has a small impact on the lookup rate. We expected that the reduction of the average path length brought about by in-network content would have lowered the number of nodes involved in transferring a content, and the number of NRS lookups.

Thus, we analyzed the ICN network traffic, and found out that a content caching strategy based on chunks, like the one we (and CCNx) are using, may reduce the potential benefit of in-network content caching on the lookup rate.

In fact, even though the probability of finding a single chunk in the cache may be high, the probability of finding all the chunks of a given content (122 in our workload) in the cache is rather small. If only a single chunk of a complete content is not found in a cache of a node, that node will require to forward the Interest message and this may produce an NRS lookup. The same situation may occur if no chunk is stored in the content cache of the node; when the first Interest message is received, the node may perform an NRS lookup and cache the ICN route in the FIB. The FIB will then be used to forward other Interest for other chunks of that content. Thus, in both cases of single-chunk-cache-miss and no-chunk-in-the-cache a single lookup may be executed. This explains the low impact of content caching on the lookup rate.

This result suggests a future work consisting in analyzing in-network content caching mechanisms that cache the whole content, rather than chunks of it.

Fig. 11 shows the amount of total traffic exchanged by ICN nodes during the test, versus the size of the content cache size of the nodes, in case of a FIB size equal to 100. As already stated in [24], we find out that the increase of the content cache size yields a decrease of the network traffic that eventually follows a logarithmic decay. This logarithmic behavior implies that the en-route caching technique has a benefit-to-cost ratio that is good for small caches, whose lookup table can be implemented with off-the-shelf hardware. Conversely, if we want to deploy very big caches, not only are they very expensive, but the performance improvement is relatively small. A way to further reduce traffic is to complement en-route caching with pre-fetching techniques, à la CDN.

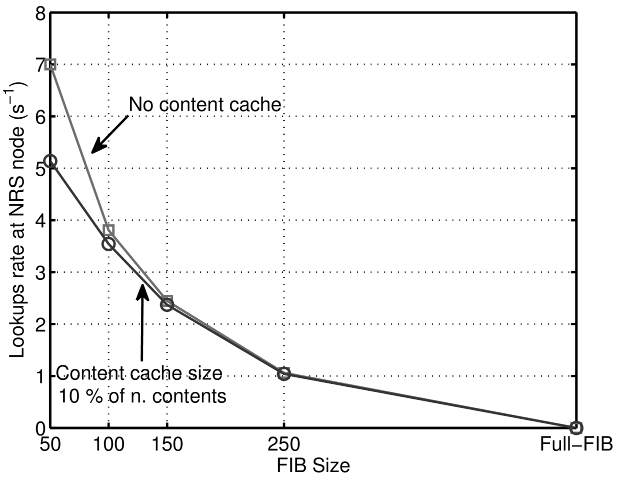


Fig. 10. Lookup rate measured at the NRS node versus FIB size

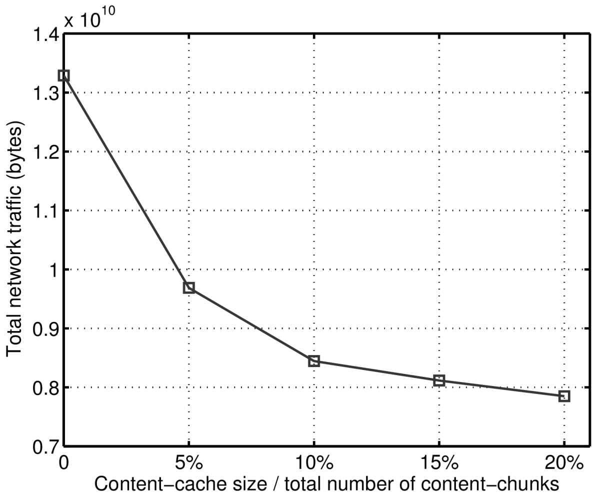


Fig. 11. Total network traffic versus content-cache size, in case of a FIB size equal to 100

7 Conclusions

The Information Centric Networking paradigm poses several technical challenges. Among them, an important one concerns the scalability of its, distinctive, routing-by-name functionality. To evaluate the scalability of routing we must consider two different issues. The first one concerns the size of the routing tables; the second one concerns the rate of routing message updates [45].

In this chapter we presented a proposal for a Lookup-and-cache architecture, which copes with the first scalability issue. The design of an ICN routing protocol that limits the rate of routing messages remains an "orthogonal", open issue, which we leave to further studies. Indeed, our Lookup-and-Cache architecture does not impose the use of a specific ICN routing protocol to exchange routing entries among NRS nodes. For instance name-based versions of BGP [41] or OSPF [42] could be viable candidates.

We used simulations and experiments over OneLab: i) to show that our Lookup and Cache architecture is feasible with current memory technology, and ii) to evaluate its performance. Our findings are that: i) it is necessary to carefully dimension the path from ICN nodes to the NRS, otherwise the lookup delay can become significant. However, this delay has to be compared with the current DNS resolution delay, as far as user perceptions are concerned, so it does not appear that this is a very limiting factor; ii) in-network *content-caching* based on chunks does not seem very useful in reducing the lookup rate; it could be worthwhile to analyze mechanisms that cache the whole content rather than chunks of content; iii) the rate of improvement of en-route *content caching* as a function of the content cache size is not very steep; this suggests to explore also other caching strategies, à la CDN.

Finally, it is worth to note that the Lookup and Cache architecture is very much in agreement with the so-called Software Defined Networking (SDN) paradigm. In

SDN, the network control plane is implemented in a dedicated device, which remotely controls packet switches providing data plane functionality. Indeed, we are implementing the Lookup and Cache architecture in the OpenFlow framework [46][47], which is a popular implementation of the SDN concept.

References

1. Cheriton, D., Gritter, M.: TRIAD: a scalable deployable NAT-based internet architecture. Technical Report (2000)
2. Koponen, T., Chawla, M., Chun, B.G., Ermolinskiy, A., Kim, K.H., Shenker, S., Stoica, I.: A data-oriented (and beyond) network architecture. In: Proc. of ACM SIGCOMM 2007, Kyoto, Japan, August 27-31 (2007)
3. Jacobson, V., et al.: Networking named content. In: Proc. of ACM CoNEXT 2009, Rome, Italy, December 1-4 (2009)
4. Smetters, D., Jacobson, V.: Securing Network Content. PARC technical report (October 2009)
5. Trossen, D., Sarela, M., Sollins, K.: Arguments for an information-centric internetworking architecture. SIGCOMM Computer Communication Review 40, 26–33 (2010)
6. Oueslati, S., Roberts, J., Sbihi, N.: Ideas on Traffic Management in CCN, Information-Centric Networking. Dagstuhl Seminar
7. Detti, A., Blefari Melazzi, N., Salsano, S., Pomposini, M.: CONET: A Content Centric Inter-Networking Architecture. In: ACM SIGCOMM Workshop on Information-Centric Networking (ICN 2011), Toronto, Canada, August 19 (2011)
8. Detti, A., Salsano, S., Blefari Melazzi, N.: IPv4 and IPv6 Options to support Information Centric Networking. Internet Draft, draft-detti-conet-ip-option-02, Work in progress (October 2011)
9. Salsano, S., Detti, A., Cancellieri, M., Pomposini, M., Blefari Melazzi, N.: Transport-layer issues in Information Centric Networks. In: ACM SIGCOMM Workshop on Information-Centric Networking (ICN 2012), Helsinki, Finland, August 17 (2012)
10. Blefari Melazzi, N., Detti, A., Pomposini, M., Salsano, S.: Route discovery and caching: a way to improve the scalability of Information-Centric Networking. In: IEEE Global Communications Conference 2012 (Globecom 2012), Anaheim, California, December 3-7 (2012)
11. SAIL project website, `http://www.sail-project.eu/`
12. PURSUIT project website, `http://www.fp7-pursuit.eu`
13. COMET project website, `http://www.comet-project.org/`
14. Named-Data Networking (NDN) project website, `http://named-data.org/`
15. COAST project website, `http://www.coast-fp7.eu/`
16. CONVERGENCE project website, `http://www.ict-convergence.eu`
17. OFELIA project website, `http://www.fp7-ofelia.eu/`
18. OneLab website, `http://www.onelab.eu/`
19. ISO/IEC 21000-2 – Information technology – Multimedia framework (MPEG-21) – Part 2: Digital Item Declaration
20. Chiariglione, L., Difino, A., Blefari Melazzi, N., Salsano, S., Detti, A., Tropea, G., Anadiotis, A.C.G., Mousas, A.S., Venieris, I.S., Patrikakis, C.Z.: Publish/Subscribe over Information Centric Networks: A Standardized Approach in CONVERGENCE. In: Future Network & Mobile Summit 2012, Berlin, Germany, July 4-6 (2012)

21. ISO/IEC 23006 – Information Technology – Multimedia Service Platform Technologies (MPEG-M)
22. Detti, A., Pomposini, M., Blefari Melazzi, N., Salsano, S.: Supporting the Web with an Information Centric Network that Routes by Name. Elsevier Computer Networks 56(17), 3705–3722
23. Kuzmanovic, A., Knightly, E.W.: Receiver-Centric Congestion Control with a Misbehaving Receiver: Vulnerabilities and End-point Solutions. Elsevier Computer Networks 51, 2717–2737 (2007)
24. Ghodsi, A., Koponen, T., Raghavan, B., Shenker, S., Singla, A., Wilcox, J.: Information-Centric Networking: Seeing the Forest for the Trees. In: Proc. of the 10th ACM Workshop on Hot Topics in Networks (HotNets-X), Cambridge, Massachusetts, November 14-15 (2011)
25. Zhao, X., Pacella, D.J., Schiller, J.: Routing Scalability: An Operator's View. IEEE Journal on Selected Areas in Communications 28(8) (October 2010)
26. Trotter, G.: Terminology for Forwarding Information Base (FIB) based Router Performance. IETF RFC 3222
27. Meyer, D., Zhang, L., Fall, K.: Report from the IAB Workshop on Routing and Addressing. IETF RFC 4984
28. Perino, D., Varvello, M.: A Reality Check for Content Centric Networking. In: ACM SIGCOMM Workshop on Information-Centric Networking (ICN 2011), Toronto, Canada, August 19 (2011)
29. BGP Routing Table Analysis Report, `http://bgp.potaroo.net`
30. Cisco Carrier Routing System, `http://www.cisco.com/en/US/prod/collateral/routers/ps5763/prod_brochure0900aecd800f8118.pdf`
31. Juniper T Series Core Routers, `http://www.juniper.net/elqNow/elqRedir.htm?ref=http://www.juniper.net/us/en/local/pdf/datasheets/1000051-en.pdf`
32. Breslau, L., et al.: Web Caching and zipf-like Distribution: Evidence and Implications. In: Proc. IEEE INFOCOM, New York, NY, USA, March 21-25 (1999)
33. Alexa Web Information Company, "Top 1,000,000 Sites", `http://s3.amazonaws.com/alexa-static/top-1m.csv.zip`
34. Jung, J., Sit, E., Balakrishnan, H., Morris, R.: DNS Performance and the Effectiveness of Caching. IEEE/ACM Transactions on Networking 10(5) (October 2002)
35. CAIDA Internet Trace Storage, `https://data.caida.org/datasets/passive-2010/`
36. MAWI Working Group Traffic Archive, `http://mawi.wide.ad.jp/mawi/samplepoint-F/2011/`
37. `http://netgroup.uniroma2.it/Andrea_Detti/Lookup-and-Cache/Feasibility-check/traces/`
38. CCNx project web site, `http://www.ccnx.org`
39. `http://netgroup.uniroma2.it/Andrea_Detti/Lookup-and-Cache-OneLab/lc_onelab.tar.gz`
40. GEANT Pan-European research network, `http://www.geant.net/`
41. Narayanan, A., Previdi, S., Field, B.: BGP advertisements for content URIs, INTERNET-DRAFT draft-narayanan-icnrg-bgp-uri-00 (July 2012)
42. Wang, L., Mahmudul Hoque, A.K.M., Yi, C., Alyyan, A., Zhang, B.: OSPFN: An OSPF Based Routing Protocol for Named Data Networking. NDN Technical Report NDN-0003 (July 2012)

43. Dan, A., Towsley, D.: An approximate analysis of the LRU and FIFO buffer replacement schemes. SIGMETRICS Perform. Eval. Rev. 18, 143–152 (1990)
44. Detti, A., Pomposini, M., Blefari Melazzi, N., Salsano, S., Bragagnini, A.: Offloading cellular networks with Information-Centric Networking: the case of video streaming. In: IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2012, WoWMoM 2012 (2012)
45. Elmokashfi, A., Kvalbein, A., Dovrolis, C.: On the scalability of BGP: the roles of topology growth and update rate-limiting. In: Proc. of ACM CoNEXT 2008, Madrid, Spain, December 9-12 (2008)
46. Blefari Melazzi, N., Detti, A., Morabito, G., Salsano, S., Veltri, L.: Supporting Information-Centric Functionality in Software Defined Networks. In: IEEE International Conference on Communications (ICC 2012), Ottawa, Canada, June 10-15 (2012)
47. McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J.: OpenFlow: Enabling Innovation in Campus Networks. White paper (March 2008), http://www.openflow.org