III. The Network Level (CONET)

Andrea Detti, Stefano Salsano and Nicola Blefari Melazzi

Electronic Engineering Department, University of Rome "Tor Vergata", Rome, Italy

Abstract

CONET (Convergence Network) is the network-layer of the CONVERGENCE project. It is an Information Centric Network, which extends the CCNx one in several aspects, including routing scalability, transport mechanisms, security handling, integration with IP, etc. This section describes services and functionalities of CONET and reports some performance evaluations, carried out through laboratory and PlanetLab test-beds.

3.1 Introduction

Network level functionalities of the CONVERGENCE system are provided by an Information-Centric Network (ICN), named CONET (Convergence Network).

The ICN paradigm envisages a network-layer thoroughly meant for information dissemination, rather than for point-to-point transfers of raw bits (Cheriton & Gritter, 2010; Koponen et al., 2007; Jacobson, et al., 2009; Smetters & Jacobson, 2009; Trossen et al., 2010; Detti et al. 2011). In ICN, the network layer provides users with named content, instead of communication channels between hosts. The basic functions of an ICN infrastructure are to: i) address content by adopting an addressing framework based on names, without a reference to the current content location (i.e., location-independent names); ii) route a user request, based only on the content-name, towards the "closest" location containing the required content; potential locations include not only the origin server of that content but also network caches or even devices of other users that downloaded the same content beforehand; iii) deliver the content back to the requesting host.

The main ICN benefit is a simplification of design, deployment and management of content distribution services. In this sense, an ICN does not extend the set of end-user services that an IP stack could provide; rather simplifies the business of content and service providers.

Currently, major content and service providers "patch" the inefficiency of IP data dissemination, by using dozens of custom extra-IP functionalities, e.g. HTTP

proxies, Content Delivery Networks, multi-homing, multicast delivery, etc. The drawback of such heterogeneous deployment is the burden of achieving an efficient interplay among several functionalities, in case offered by different company. This tune up is critical and complex, not only from a technical point of view but also from a management one.

An ICN is directly meant for information dissemination. Consequently, an ICN relieves providers from arranging extra-ICN functionalities. For instance data replication, caching, multi-homing and multicast delivery are inner ICN functionalities, directly handled by the network layer, so simplifying network design, deployment and management.

Through an ICN Application Programing Interface (API), a user may request by-name an information item and the ICN network-layer provides the user with the item, fetching it from a serving device that the network-layer has autonomously selected. Serving device could be the original server publishing the information, a replica of the original server or a cache of a network device or even a device of another user that downloaded the same information item beforehand.

Through an ICN API a content-provider may publish by-name an information item and the ICN network-layer properly configures its routing plane so that requests of the published item will be served.

An ICN is aware of distributed information, since the network-layer identifies each information item with a unique name-based identifier, and uses data units that include such name-based identifiers. ICN data-units may also convey request of information items. In both cases, a node is aware of "what" a data-unit refers to. This awareness is the enabler of content-based functionalities such as: i) routing-by-name, ii) caching; iii) support for mobile, multicast and peer-to-peer communications; iv) support for time/space-decoupled model of communications; v) content-oriented security model; vi) content-oriented quality of service, access control and traffic engineering.

From theory to practice, the design of an ICN architecture sets up several technical challenges, as for instance:

Primitives & interfaces - define the relationship of the ICN protocols with the overall architecture, including their positions and connections with IP and current protocols; for instance, will ICN be the new narrow waist of the Internet (i.e. the lowest-level global network primitive, the only way to establish global communication) or will it sits over IP? Other important issues of this component include the basic primitives and interfaces, e.g., does the ICN layer offer a publish/subscribe primitive or is this functionality delegated to upper layers? How can we change/extend the socket API, which is one of the main responsible of the current "ossification" of the Internet?

naming scheme – the naming-scheme specifies the identifiers for the information addressed by the ICN. The choice of a naming-scheme impacts different aspects an ICN, including the handling of name uniqueness and trademarking, routing scalability, flexibility of supporting different applications, usability, security. For instance, human-readable flat-names improve usability; hierarchical names foster aggregation of names in name-based forwarding tables; names containing the public key of the owner of the content simplify security.

name resolution – a node routes by-name a request of an information item towards a selected serving device. Hence, the ICN node should resolve the name of requested items in the "physical" address of next ICN node towards the selected serving device. Several name-resolution approaches are possible, ranging from an off-path resolution, e.g. based on DHT, to an en-route hop-by-hop resolution exploiting name-based routing tables.

routing scalability - with respect to IP, the routing plane of an ICN has to handle a number of information items and corresponding names that is much bigger than the number of IP network prefixes. This has implications on the size of ICN routing tables, on the complexity of lookup functions and on the distribution of ICN routing information and is one of the main concerns of ICN.

information delivery – an ICN addresses information items rather than hosts, so it needs a technique to route back the requested information item from the serving device to the requesting device. Some architectures propose to use plain IP means, other ones propose to face the issue through ICN's own means.

segmentation mechanisms – these mechanisms are needed to split an information item or a content in different chunks (each chunk is an autonomous data unit with embedded security and addressable by the routing plane). Content to be transported over an ICN can be very variable in size, from few bytes to hundreds of Gigabytes. Therefore it needs to be segmented in smaller size data units, typically called chunks, in order to be handled by ICN nodes. A chunk is the basic data unit to which caching and security is applied.

transport mechanisms – as regards the transport protocol, we favour a receiver-driven approach: in ICN the transport of an information item does not exploit an end-to-end session and while the requesting device remains the same, the serving device may change also on a chunk-by-chunk basis. This requires a complete rethinking of actual Internet transport mechanisms towards a receiver-driven approach, where the whole transport logic is on the receiver side. Serving applications split information items in chunks and assign unique names to chunks. On the other side, application clients fetch sequentially these chunks, according to a receiver-driven transport algorithm. As in IP, transport mechanisms should be tailored to application characteristic, e.g. file download, video streaming, voice over ICN, etc.

in-network caching – ICN nodes may cache chunks of information. Differently from traditional HTTP caching, an ICN is a *cache network*; this implies the need of properly devising a replication strategy that optimizes the caching space, e.g. by avoiding excessive duplication of content in the network caches.

security and privacy challenges - security and privacy issues in ICN tackle several aspects: i) integrity: received content has not been modified, i.e. it is the originally published one; ii) provenance: source of the content is authentic, i.e. the data is provided by the original creator; iii) relevance: received content is really the content requested by the user. The verification of these criteria should be done not only at the receiver side, but also in network nodes, as it is important that the network be protected from pollution of content-caches with fake information items. In addition, the network should protect information consumers from profiling or censorship of their requests. These issues, extensively addressed in traditional networks, require a significant rethinking when challenged against the unique distinguishing characteristics of ICNs. Traditional network security protocols such as IPsec or TLS focus on protecting the communication between an information consumer and a content server, and do this by deploying trustworthy infrastructures devised to enforce authentication and access control primitives on dedicated servers. In ICN, the requested content is not anymore associated to a trusted server or an endpoint location, but it can be retrieved from, say, a network cache managed by an hardly trusted administrative domain. This calls for data-centric security and privacy solutions, being hardly viable a secure infrastructure which involves storage servers and network caches in heterogeneous non-collaborative domains. The data-centric security model increases the communications overhead with respect to traditional IP-based solutions, where security related information are exchanged only one time per information transfer, i.e. at the start of the end-to-end session. Furthermore, the architectural binding of security and network layer functionality has to be carefully designed. In fact, it may impose severe deployment limitations. For instance, an ICN architecture based on digital certificate may not properly operate without a PKI, thus making difficult to realize self-forming ICN networks. Another example regards the denial of service due to the presence of fake content in the cache, caching should be secure, i.e. node must verify the validity of cached contents. This operation should be carefully devised to operate as much as possible with a time scale close to the line rate of nodes. Otherwise only a limited set of forwarded information items can be cached by a node.

push services – an ICN is primarily meant to enable clients to “pull” information items. However, today, several Internet services are customized to the user and these services require that client “pushes” information to server. For instance home banking, trading on-line, dynamic web servers, belong to this class of services. Therefore, an ICN architecture aiming at being the narrow waist of the Future Internet has to support also a push service model.

smooth migration path – an ICN architecture should be usable and deployable in a scalable way, i.e. should support a smooth migration path from current applications and networks technology based on TCP/IP, to ICN applications and networks.

In this section, we describe the Information Centric Network of the Convergence project, namely CONET (Detti et al., 2011, Detti et al., 2012). In short, we can say that CONET:

1. uses a hierarchical naming scheme;
2. performs name-resolution on-path, by using name-based routing table;
3. copes with the routing scalability issue by using our proposed Lookup and Cache routing architecture;
4. delivers information to requesting user either by means of control data temporary left in network nodes, or inserted in network data units;
5. implements an efficient receiver driven TCP-like transport algorithm for information delivery services;
6. uses Identity Base Signature and/or self-certifying names to carry out security related operations;
7. provides in-network secure caching;
8. supports push services by our proposed named-sap concept;
9. can be deployed with an IP overlay or a clean slate approach or with our proposed integration approach (see Detti et al. 2011, Detti et al. 2011b).

Some of these features (1, 2, and 4) are derived from the CCNx architecture (Jacobson et al., 2009); the remaining ones are CONET's own. In what follows, we describe CONET in details.

3.2 Integration of the CONET network layer in the CONVERGENCE system

Figure 1 describes the architecture of the CONVERGENCE system. We have publishers that wish to provide customers with their information or services (e.g. text, picture, movie, home bank access, etc.). Information or services are described by a set of metadata that are embedded in the *middleware* data-unit, namely the VDI. The VDI can either contain also the actual information item or contain only a reference to it, where the reference is a network-layer identifier.

VDIs are exploited by middleware *engines* for different aims; engines are middleware technologies/protocols carrying out specific tasks. For instance, the middleware can exploit VDIs to offer content-based publish-subscribe services

(Chiariglione et al., 2012; Eugster et al., 2003) as follows: applications express their interests; the interest is stored by the middleware and when a corresponding information item or service is published, the middleware returns the matching VDIs; at this point, the application can exploit the CONET to either fetch the desired information item or interact with a server providing the desired service; this is possible since the VDI contains the network-layer reference to the information item or to the service.

However, publish-subscribe is not the only possible interaction model. For instance, an application could also use a request-response model in which a search engine provides the user with the VDI matching her request.

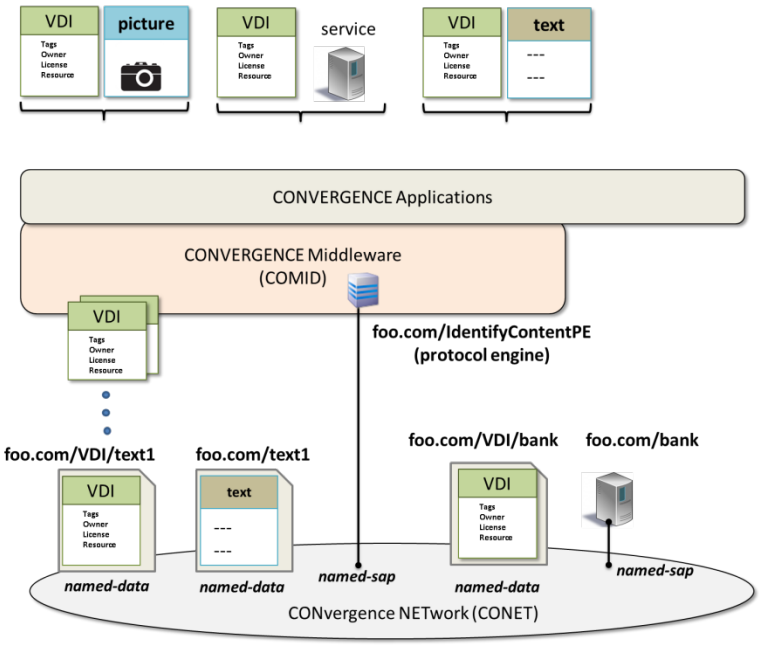

Figure 1: Integration of CONET within the CONVERGENCE system

The VDI and the related information item or service are accessed through the CONET as *named-resources*. A named-resource can either be an information item or a service, addressed by the CONET by means of a NEtwork Identidier (NID), which is a name.

Named-resources that refer to information items are called *named-data*. **Figure 1** reports a case where a text file and its associated VDI are disseminated in the CONET as two named-data items. The named-data item related to the text file has the name (i.e., a NID) "foo.com/text1", and the named-data item related to the VDI has the name "foo.com/VDI/text1".

Named-resources that refer to services are called named-service-access-point, briefly *named-sap*. A named-sap is merely a "port" towards an upper layer entity addressed by the CONET through a name. The use of named-saps enables CONET to support *interactive* services, i.e. services that require a custom client/server interaction to provide personalized contents or actions. Examples of this interactive service class include: services offered by Protocol Engines (PEs) of the CONVERGENCE middleware (COMID), HTTP servers of dynamic Web pages, home banking or trading on-line HTTP servers and SMTP servers.

Figure 1 reports also a case where a home banking service and its associated VDI are disseminated in the CONET as a named-sap and a named-data item, respectively. The named-sap of the home banking service has the name "foo.com/bank", and the named-data of the VDI has the name "foo.com/VDI/bank". The figure reports a named-sap associated with a COMID IdentifyContent PE, which is addressed by the CONET through the "foo.com/IdentifyContentPE" named-sap.

3.3 CONET services

3.3.1 Publication of named-resources

The CONET enables users to publish and to revoke named-resources. A resource can be replicated in different geographical locations by using the same name.

Figure 2 reports the case of a user that publishes both a named-data item and a named-sap. The named-data of **Figure 2** is a text file identified by the CONET with the name "foo.com/text1". To publish the named-data, the user exploits the API provided by a CONET Serving Node (SN) to *store* the named-data item in a local Repository, and to *advertise* the presence of "foo.com/text1" on the CONET routing plane.

The named-sap of **Figure 2** is a home banking service provided by a Server and identified by the CONET with the name "foo.com/bank". To publish the named-sap, the user exploits the API of the Serving Node to *advertise* the reachability of the service "foo.com/bank" on the CONET routing plane.

Albeit not reported in **Figure 2**, the user may exploit the CONET API to remove the text file from the repository and to withdraw the identifier "foo.com/text1" from the CONET routing plane. A similar action may occur in case of revocation of the home banking named-sap.

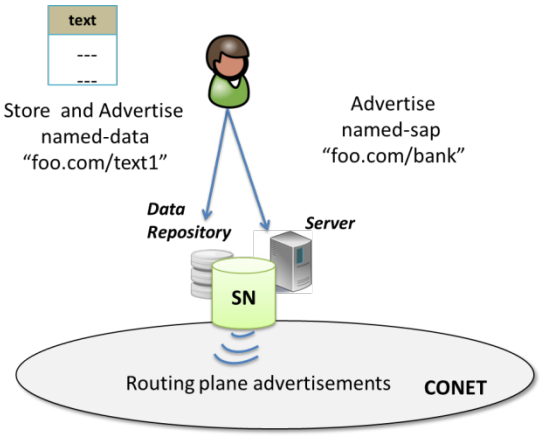

Figure 2: Publication of named-resources

3.3.2 Access to named-resources

CONET provides users with the possibility of accessing named-resources by using their network identifiers (NIDs), i.e. their names. When a named-resource is a named-data item, the CONET delivers it to intended recipients. When a named-resource is a named-sap, the CONET provides the means to exchange information between a requesting upper layer entity and the upper layer entity addressed by the named-sap.

Figure 3 depicts the access to named-resources provided by the CONET. In case of the named-data item “foo.com/text1”, the network *routes-by-name* the request of “foo.com/text1” towards the best Serving Node that has the item. The Serving Node provides the named-data item, and the CONET sends it to the recipient. As we will see in the next section, a CONET node may cache named-data items, therefore the named-data “foo.com/text1” item could also be sent to the recipient by an en-route node, rather than from the original Server.

Figure 3 also reports the access to a named-sap. In this example, the user sends her credential (“usr:pwd”) towards a remote named-sap, whose name is “foo.com/bank” and gets back a response from the Server.

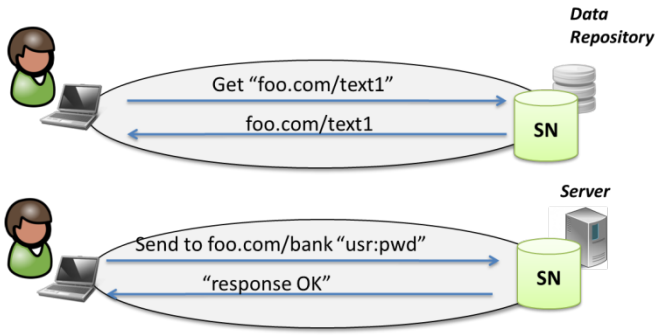

Figure 3: Access to named-data and named-sap

3.4 Naming model

To identify named-resources, the CONET uses a hierarchical naming scheme, formed by a *PrincipalId* and a *Label*, i.e. *PrincipalId/Label*. PrincipalId is a string, e.g. “foo.com”, that uniquely identifies the principal of the resource. Label, e.g. “text1”, is an identifier that uniquely identifies a resource among those published by a principal. To make a comparison with Internet URL, the PrincipalId is the domain-name and the Label is the Path.

In general, both the PrincipalId and the Label are formed by name-components that are strings separated by the “/” character. For instance, the identifier “foo.com/content/doc/text1” is formed by foo.com as PrincipalId and “content/text/text1.txt” as Label, and this Label has three components: “content”, “doc”, and “text1”.

A PrincipalId could be a human-readable name or the public key of the principal. In the latter case, we have a so called *self-certifying name* (Koponen et al., 2007). Therefore CONET supports both human-readable and self-certifying names.

3.5 Data Model

3.5.1 Data model for named-data related services

Figure 4 reports the data model used by the CONET for named-data. An information item is linked with a unique network identifier, e.g. “foo.com/text1”, so as to form a named-data.

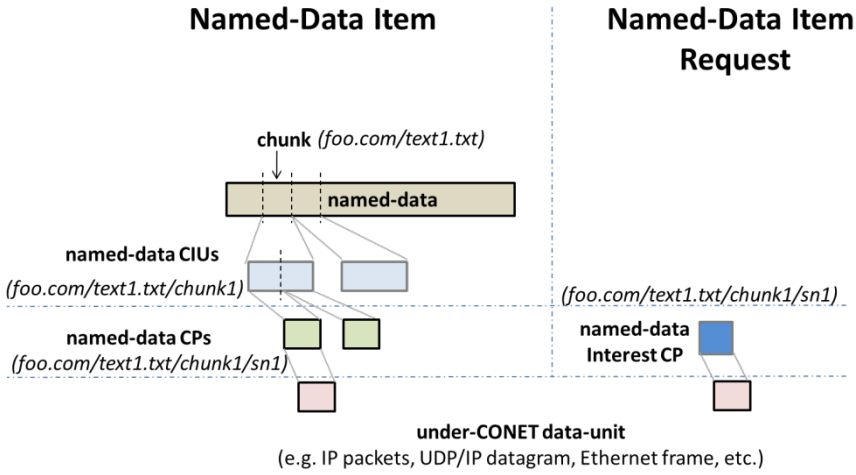

Figure 4: Data model for named-data services

A named-data is segmented in different chunks and each chunk is packaged in a data unit called *named-data* Content Information Unit (CIU). A named-data CIU is uniquely identified by the network with the name of the whole named-data item and an identifier of the chunk number. The identifier of the chunk number can either be included in the name, e.g. “foo.com/text1.txt/chunk1”, or included in a suitable field of the CIU header (Detti et al., 2011). A named-data CIU contains security information (Smetters & Jacobson, 2009), so that traversed nodes may securely cache it, by avoiding denial of services attacks due to caching of fake contents. Thus, the named-data CIU is the caching data-unit of the CONET.

To reduce the security overhead and the rate of security checks performed by caching nodes, the size of the named-data CIU should be greater than the usual payload transported by IP packets. For instance, reasonable size of the named-data CIU could be in the order of tens or hundreds of IP packets / Ethernet frames. For this reason, a named-data CIU is further segmented into so-called named-data *carrier packets* (CP), whose size fits the end-to-end maximum transfer unit. A named-data CP is uniquely identified by the network with the name of the named-

data CIU plus a segment number; also in this case, segment number can either be included in the name, e.g. “foo.com/text1.txt/chunk1/sn1”, or included in a suitable field of the CP header (Detti et al., 2011).

As we will see in the next section, applications download a named-data by sequentially downloading all CIUs and, hence, all CPs. To request a named-data CP, a user issues a named-data Interest CP[1], which includes the identifier of the desired carrier packet (see Figure 4).

3.5.2 Data model for named-sap related services

Figure 5 reports the data model used by the CONET for named-sap related services.

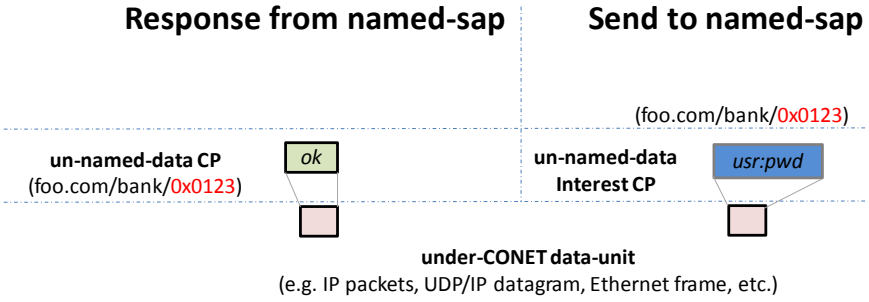

Figure 5. Data model for named-sap services

A named-sap is the coupling between a server upper layer entity and a network identifier, i.e. a name. For instance Foo’s server for home banking could be addressed by the CONET through the named-sap “foo.com/bank”.

A client upper layer entity interacts with a server upper-layer entity by exchanging data that are of exclusive interest of such client-server interaction and, hence, are *un*-cacheable for further re-use. For this reason, we refer to this type of data as *un*-named-data.

Client data (e.g., “usr:pwd”) are sent to server through data units called un-named-data Interest CPs. These messages include the network identifier of the named-sap and a nonce, used to identify the specific *request-response* interaction; e.g. “foo.com/bank/0x0123”, where “foo.com/bank” is the identifier of the named-sap and “0x0123” is the nonce.

[1] We note that in [6], we denoted this named-data Interest CPs with another name: Interest CIU. We changed the name because an Interest CIU is directly mapped in an underlying carrier-packet.

Server responses are sent back within un-named-data CP, which includes the network identifier of the named-sap and the nonce of the Interest CP.

3.6 Service interaction model

This section describes the CONET service interaction model for named-data and named-sap related services, respectively.

3.6.1 Interaction model for named-data pull services

Figure 6 (left) depicts the end-to-end interaction for the fetching of a named-data. An end-node downloads a named data by retrieving all its named-data CIUs and, hence, CPs. To download a named-data CPs, a client sends out a named-data Interest CP (briefly, 'data CP' in the figure), which contains the identifier of the desired named-data CP, e.g. "foo.com/text1/chunk1/sn1". The CONET routes-by-name the Interest message toward the Serving Node, which sends back the requested data within a named-data CP (briefly, 'data CP' in the figure). To download the whole set of named-data CPs, the client may adopt a TCP-like receiver-driven approach (Salsano et al., 2012; Kuzmanovic & Knightly, 2007), working as follows: i) TCP ACKs are replaced by Interest CPs; ii) TCP segments are replaced by named-data CPs; iii) traffic control operation is carried out at the receiver side, by using a TCP-like congestion window (cwnd) control, applied on the number of in-flight Interest CP messages.

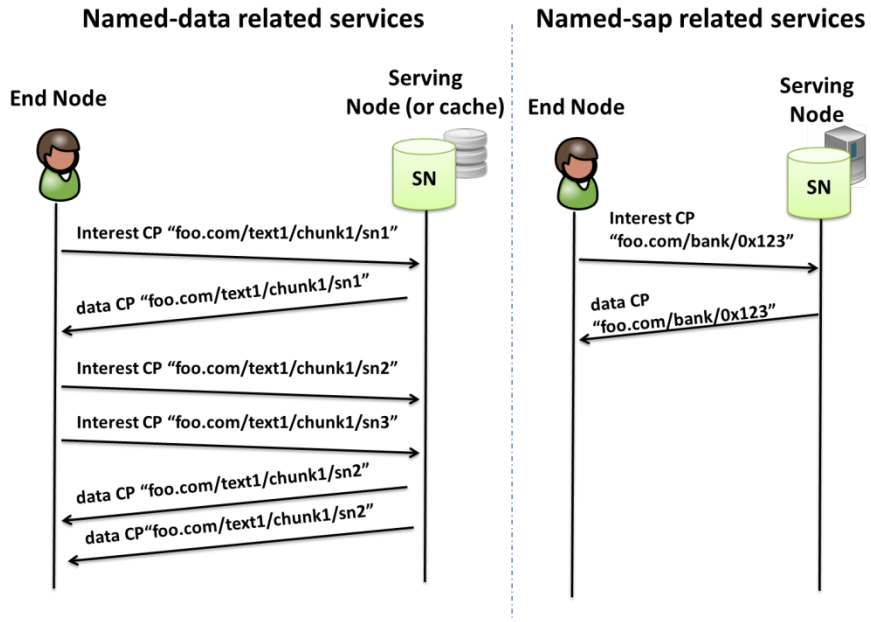

Figure 6: End-to-end interaction models for named-data (left) and named-sap (right) services

We observe that the use of a receiver-driven TCP-like transport algorithm is suitable for download services, while in the case of e.g. live streaming (A. Detti, et al., 2012c) or voice (Jacobson et al., 2009b), different receiver-driven flow control mechanisms should be designed, according to the specific application requirements.

3.6.2 Interaction model for named-sap push services

Figure 6 (right) depicts the end-to-end interaction between an end-node and a remote server behind a named-sap. In this case the interaction model is a request-response, as previously described in section 3.5.2.

3.7 Architecture and functions

Figure 7 reports the CONET network architecture. CONET nodes are interconnected by "sub-systems" that 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 IP hosts and routers can be connected via different layer 2 technologies. A CONET sub-system may include: i) CONET end-nodes that access named-resources; ii) CONET Serving-nodes that provide named-resources and iii) CONET nodes that relay carrier-packets between sub-systems and optionally cache named-data CIUs.

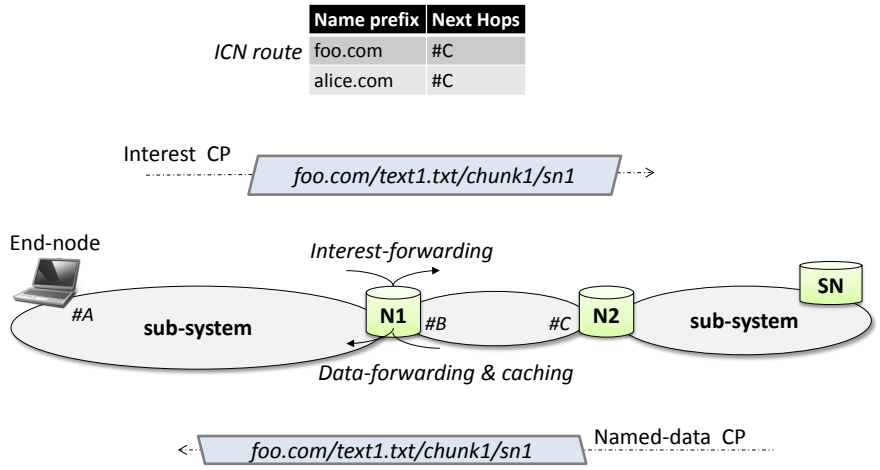

Figure 7 – Network architecture

Routing (by-name) of Interest - To route an Interest CP, a generic CONET node uses a name-based Forwarding Information Base (FIB), whose entries are in the form <name-prefix, next hops>. A longest matching algorithm, based on characters, selects the best entry of the FIB. For instance, in case of Figure 7, when node N1 receives an Interest CP for "foo.com/text1/chunk1/sn1", then the longest match selects the FIB entry "foo.com" (see the FIB reported in the top of the figure). Then node N1 forwards the Interest CP towards the next CONET interface of the path, i.e. interface #C of N2.

The whole set of possible name-prefixes could be too large to be stored in the "fast" FIB memory, e.g. based on SRAM or TCAM technology. To overcome this aspect of routing scalability, we propose that (A. Detti et al., 2012; N. Blefari Melazzi et al. 2012b):

- the PrincipalId is the *longest* routing prefix; i.e. name-prefixes contained in the routing table distributed at the inter-domain level either are PrincipalIds or are

shorten than PrincipalId, in order to aggregate more PrincipalIds in a same routing entry. This rule has to be applied at least at the inter-subsystems (or inter-domain) level;
- the FIB is used as a *route cache* (Changhoon et al., 2009), to temporarily store the limited set of routes necessary to support the on-going end-to-end interactions. We call these routes *active routes*.

On the base of real Internet traces and in a case where PrincipalIds are the actual 2 10^8 Web domain-names, in (Detti et al., 2012) we show that the set of active routes is relatively small (10^4), both with respect to the whole set of possible routes (2 10^8), and also with respect to the current storage capacity of SRAM memory technology (10^6 name prefixes) (Zhao et al., 2010; Perino & Varvello, 2011).

In case the FIB does not contain an entry to route-by-name an incoming Interest CP, the node lookups the routing entry in a Routing Information Base (RIB), deployed in a centralized Name Routing System (NRS) node, which hosts the CONET Routing Engine. A RIB may be implemented through a bank of "slow" DRAM memories, whose overall memory space is able to store the whole set of possible name-prefixes. We call such routing-by-name architecture *Lookup-and-Cache*.

Figure 8 depicts an example of Lookup and Cache operations during the forwarding of an Interest CP. When node N1 receives the Interest CP for "foo.com/text1/chunk1/sn1", the node does not have a valid routing entry in the FIB, hence lookups the entry in the remote RIB. Then it inserts the routing entry ("foo.com") in the FIB and forwards the Interest CP.

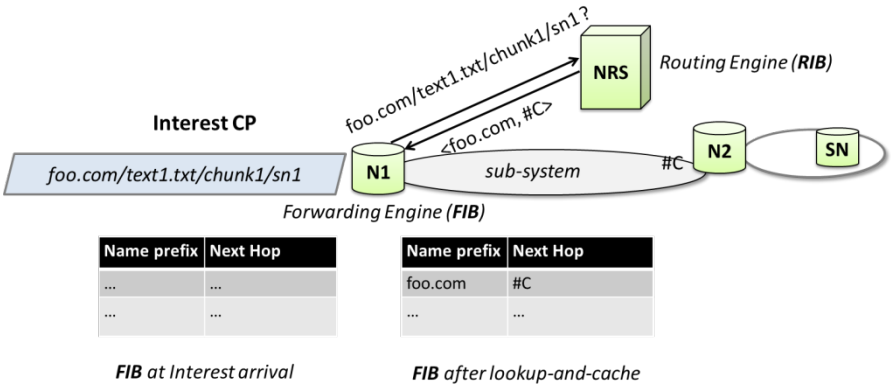

Figure 8: Lookup and Cache routing-by-name architecture, Interest forwarding

We observe that the Lookup and Cache architecture is perfectly in line with the Software Defined Network paradigm: the NRS node functionality could be im-

plemented in the central network controller (McKeown et al., 2008; Blefari-Melazzi et al., 2012).

Routing (by-name) of Data– To support *information delivery,* i.e. to route data from serving node to the requesting device, a possible, *state-full*, approach is that a node temporarily stores the couple <CP identifier, previous-hop interface list> in a Pending Information Table (PIT), during the forwarding of an Interest CP. The PIT contains information about the set of Interest CPs received by a node and not yet served, i.e. messages for which the node has not yet sent back the related named or un-named CPs. PIT entries are grouped by name and an interface list field contains the addresses of the previous-hop interfaces that forwarded the related Interest messages. For instance, in the case of Figure 7, when node N1 receives the Interest "foo.com/text1/chunk1/sn1" forwarded by the previous end-node, then node N1 inserts in the PIT the entry <"foo.com/text1/chunk1/sn1", #A>, where #A is the physical address (e.g. IP or Ethernet address) of the end-node interface.

When a node receives a named/un-named data CP, it lookups the enclosed name in the PIT, forwards the CP towards all the interfaces contained in the interface list and deletes the PIT entry. In this example, when node 1 receives a named-data CP for "foo.com/text1/chunk1/sn1", then it lookups the entry in the PIT, forwards the message towards the #A interface and deletes the PIT entry.

In-network, en-route, caching – A CONET node may cache received named-data CIUs in a local memory. Since a named-data CIU is an aggregation of named-data CPs, then caching involves a reassembly operation. Moreover, a CIU is inserted in the cache only if the enclosed security information confirms its proper validity. When an en-route node receives a named-data Interest CP, the node first checks the presence of the related named-data CP in its cache. If a cache hit occurs, the node directly sends back the named-data CP. In case of cache miss, the node executes the forwarding and PIT operations previously described.

In addition to en-route caching, other caching approach could be used, depending on the specific environment. For instance, in (Gallucio et al., 2012; Detti et al., 2012b) authors propose a caching approach based on overhearing, for satellite networks.

Routing protocol – The lookup and cache routing architecture requires a routing protocol to distribute name-prefixes and setup the RIBs of NRS nodes. The Lookup-and-Cache architecture is independent from the specific routing protocol implementation. To show an example, we implemented a simple routing protocol based on the REGISTER and UNREGISTER functions proposed by the DONA architecture (Koponen et al., 2007), adapted to our specific network model.

As show in Figure 9, a Serving Node (SN) REGISTERs and UNREGISTERs the name-prefixes (e.g., "foo.com") of "its" named-data items in the local NRS node. The local NRS node and the next ones forward the

REGISTER/UNREGISTER messages toward their parents and peers neighbours up to a NRS of the tier-1 level. As in the case of an Interest message, the routing of REGISTER/UNREGISTER messages is by-name. Indeed, each NRS node has a named-sap used to receive routing messages, e.g. “ss1.org/routing.sap”. Reception of REGISTER and UNREGISTER messages enables an NRS node to properly setup its RIB. We observe that the same NRS may serve more than one subsystems or nodes and even a whole autonomous system (i.e. a collection of subsystems administered by the same entity, as in the Internet). In this case, the NRS would have a RIB for each served node.

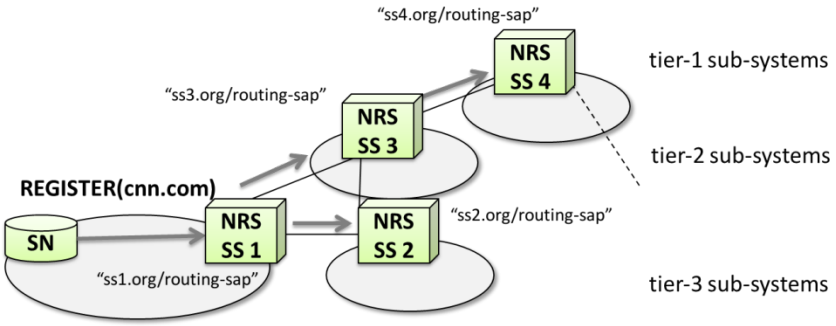

Figure 9: Lookup and Cache architecture: inter-domain distribution of name-prefixes

3.8 Security support

CONET security refers to the ability of a node to verify the validity (integrity, provenance and relevance, see Section 3.1) of named-data CIUs. The way security functionality is designed and implemented depends on the adopted naming scheme: human-readable or self-certifying names (see Section 3.4). Moreover, in order to support in-network secure caching (Ghodsi et al., 2011), named-data CIU should contain all the security information needed to verify its validity, i.e. the signature and the means to verify the signature, such as the public key of the signing principal (Detti et al., 2013).

In case of human-readable names, the CONET needs a centralized authority to control PrincipalId uniqueness. Moreover, the CONET needs a PKI infrastructure to control validity of principal’s public key. To reduce the bandwidth overhead of security, we propose to use Identity Based Signature approach, where the public key is just the PrincipalId, already contained in the name of the named-data CIU.

In case of self-certifying names, PrincipalId is the principal public key that could be randomly generated by the principal. Moreover, in this case, security re-

lated verifications do not require a PKI, as discussed in (Ghodsi et al., 2011; Detti et al., 2013). Therefore, we argue that self-certifying names are a good choice for self-forming ICNs, where the presence of a PKI may be impractical.

3.9 Migration path

The migration from services based on the TCP/IP API to services based on the CONET API requires, on the one hand, to have applications able to operate on the new API and, on the other hand, to deploy the CONET.

Regarding applications, we observe that the CONET API is quite similar to the HTTP API, which is the most used API by Web applications (Popa et al., 2010). This similarity eases the development of HTTP/CONET (or generally ICN) transparent proxies, which can be used during the migration from HTTP based application to CONET based applications. For instance, we followed this approach to design a video streaming application (Detti et al., 2012c), in which the video client (VLC) is a plain HTTP application connected to a HTTP-to-CONET proxy.

Regarding the deployment of the network, we envisage a possible migration path which starts from a first *overlay* deployment (similar to CCNx (CCNx, 2012)) where CONET uses IP as carrier and adopts dedicated hardware; to a second, more *integrated*, scenario where CONET and IP use the same data-units, and hence the same hardware.

Technically, in the first deployment scenario, CONET carrier-packets could be transported in the payload of UDP/IP packets, thus the connections among CONET nodes would be overlay links and CONET nodes would use specific hardware. In the second deployment scenario, control information of carrier-packets is integrated in the IP header, as an IPv4 options or IPv6 extension header (Detti et al. 2011, Detti et al. 2011b); in doing so, CONET nodes and IP routers could be integrated in a same hardware.

3.10 Performance analysis

In this section, we report a brief survey of the performance analysis of some CONET aspects, namely routing and transport issues. A more detailed descriptions of measurements and performance figures are contained in referenced papers.

3.10.1 Routing

In (Detti et al., 2012; Blefari-Melazzi et al., 2012b) we use real Internet traces to assess the feasibility of using the FIB as a cache of routes. We considered a deployment scenario where CONET is used to fetch current Web contents. In this case the name-prefixes handled by the CONET routing plane are the domain-names, which nowadays are in the order of 10^8. This amount is two orders of magnitude greater than the storage capacity provided by current FIBs; indeed, a SRAM memory may store a number of name-prefixes in the order of 10^6.

CONET copes with such storage limitation by using the FIB as a cache of *active* routes, i.e. of those name-based routing entries currently needed by a node to forward traffic. To prove that this approach is feasible we must show that the number of active routes is lower than the FIB storage capacity.

To verify this feasibility, we use a real Internet trace of a tier-1 Internet link (Equinix-sanjose-dirA) from which we derive a hypothetical CONET trace of a tier-1 node, as shown in (Blefari-Melazzi et al., 2012b). Then, we use the hypothetical trace to compute the number of CONET active-routes that a tier-1 node would have. Figure 10 reports the number of active-routes versus time and we observe that this number is in the order of $5x10^3$, so much lower than the FIB capacity. Therefore we can state the using the FIB as a route-cache is feasible in the assumed scenario in which COINET is used to fetch current Web contents.

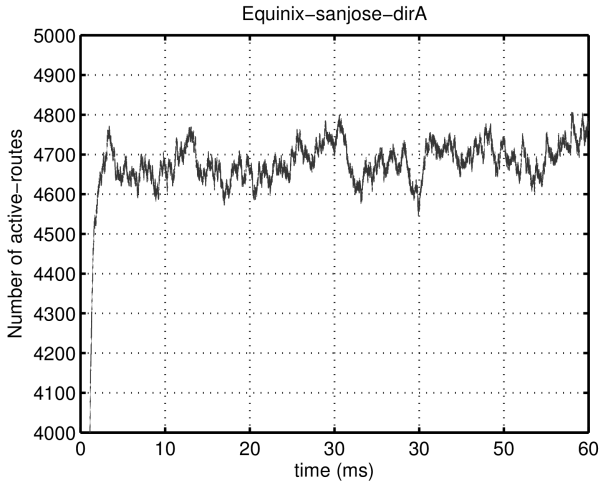

Figure 10: Number of active-routes for the Equinix-sanjose-dirA trace

In (Detti et al., 2012) we evaluate the possible performance degradation introduced by the lookup and cache routing operation, in the laboratory scenario depicted in Figure 11. We have an ICN node between two subsystems. Within sub-

system A, we have end-nodes (clients) that fetch named-data items from a serving node, located in subsystem B. The serving node publishes 10,000 named-data items and each named-data item has a different, random, name-prefix. Consequently, in this scenario a full FIB would contain such 10,000 name-prefixes. On each node, we disabled in-network caching of named-data CIUs, to avoid the influence of data caching during the routing performance assessment.

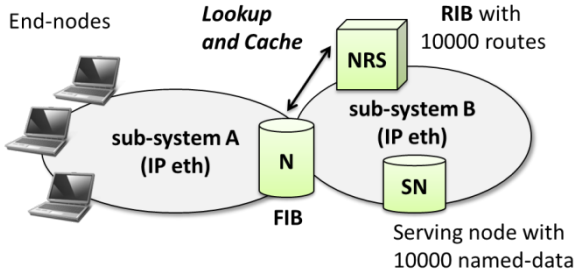

Figure 11: Lookup and Cache Testbed setup

Figure 12 reports the average download time of a named-data item versus the number of active routes. We consider three cases: in the first case we consider node N with a FIB of size100 entries, which uses the LRU replacement policy to replace cached routes; in the second case we use another replacement policy based on a Inactivity Time Out (ITO) estimator (Detti et al., 2012); in the third case we assume that node N has an unlimited FIB space, properly preloaded with the whole routing set (i.e. 10,000 routes). Results show that, as expected, when the number of active-routes is lower than the available FIB space, performance degradation introduced by a limited FIB is practically negligible, with respect to the case of unlimited FIB. Conversely, when the FIB is overloaded, i.e. the number of active-routes is greater than the FIB space, then performance degradation shows up and the ITO policy outperforms the simpler LRU.

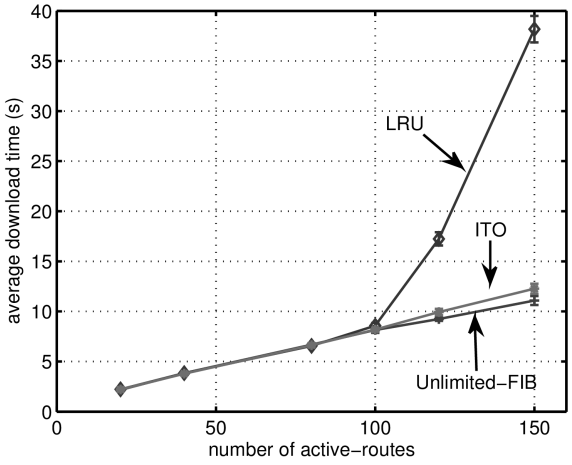

Figure 12: Average download time versus number of active-routes in laboratory test-bed

In (Blefari-Melazzi et al. 2012b), we repeated the same download delay measurement on the PlaneLab *overlay* CONET deployment shown in Figure 13, where links are UDP/IP sockets that transfer CONET carrier-packets. This scenario represents the case of a CONET *Autonomous System*, which is an aggregation of CONET subsystems under the control of the same network operator. The topology graph is generated so as to resemble the European GEANT research network. Each node is labelled with the country code name of its actual location, and serves a CONET subsystem, formed by a serving-node and an end-node (the latter is shown in the figure only in the case of the IE node). Serving-nodes publish a wide set of named-data items, whose popularity follows a Zipf distribution. End-nodes fetch named-data items, with exponential negative distributed inter-arrival times between two consecutive fetches. Lookup and Cache routing is supported by a single NRS node, which serves all nodes of the Autonomous System. Each node is connected to the NRS with a best-effort UDP/IP socket, used to perform routing lookup operations. This path is not drawn in Figure 13.

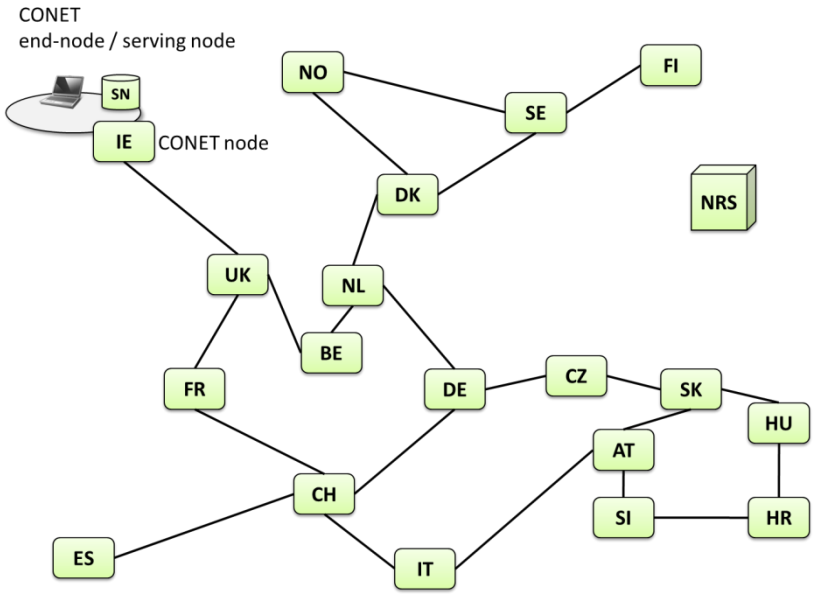

Figure 13: PlanetLab CONET overlay deployment

Figure 14 shows the average download time versus the FIB size (number of entries), comparing the case of nodes without content (named-data) cache and the case of nodes with a content cache; the content cache size (number of entries) is equal to 10% of the total number of published named-data items. The x-axis includes also an out-of-scale point, representative of a full preloaded FIB (labelled "Full-FIB") where, for each node, we use an *unlimited* FIB, pre-loaded with all name-based 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, located in the NRS node.

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

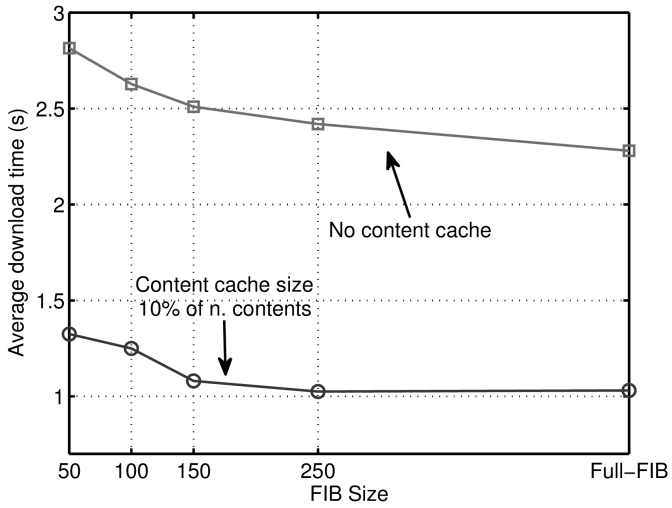

Figure 14: Average download time versus node FIB size in laboratory test-bed, with and without in-network caching

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 *lookup delay penalty* is due to the connectivity/processing delay brought about by the NRS node, which in the worst case is equal to about 350 ms. This considerable delay penalty was not present in the laboratory test-bed of Figure 12, since there we had a direct Ethernet connection between the node and the NRS. Conversely, in the PlanetLab test-bed, NRS is connected to nodes through a best-effort Internet path, which introduces a significant delay. We argue that this delay is due to the connectivity/processing delay brought about by the NRS node. Such 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.

3.10.2 Transport of named-data

As discussed in Section 3.6.1, the CONET adopts a receiver-driven protocol where endpoints exchange Interest / Named-Data CPs sequences and the exchange rate is regulated by the receiver, following principles proposed in (Salsano et al., 2012). This protocol, which we generically name Information-Centric Transport Protocol (ICTP), implements the same algorithms of TCP (slow-start, congestion

avoidance, fast retransmit, fast recovery), but adapted to the receiver-driven operation.

In (Salsano et al., 2012), we measured the performance improvement provided by CONET ICTP with respect to the transport mechanism included in the CCNx tool.

In short, the main differences between our ICTP and the transport protocol provided by the CCNx tool can be summarized as follows:

1) *Use of carrier-packets*: we can say that the data-units of the CCNx transport protocol correspond to the named-data CIUs (aka Data messages), each of which is relatively large, (e.g. 4 kB). This means they have to be segmented at the IP level. The use of such a large data-unit decreases the efficiency and promptness of congestion control. To overcome this problem, ICTP divides named-data CIUs into segments, each of which is transported by a carrier-packet, whose size is close to that of a TCP segment. In other words, the carrier-packet is the ICTP data-unit and no longer the whole named-data CIU. This suggests that the performance of the transport layer will be similar to that of the transport layer in TCP.

2) *Mimicking TCP congestion control*: CNNx implements a simple congestion control algorithm that resembles the traditional Selective-Repeat ARQ with fixed window size. Conversely, ICTP thoroughly mimics the TCP Reno congestion control algorithms.

We evaluate the performance of ICTP in a laboratory scenario with a single sub-system; in the scenario an end-node fetches named-data from a serving node with a 10Mb/s link. We compared the ICTP performances with the performances of the congestion control in CCNx. Figure 15 shows the application goodput measured for CCNx and for ICTP versus the IP packet loss probability and for two different sizes of the named-data CIUs (4 kB and 32 KB). In the ideal lossless case, the performance of the current CCNx transport implementation is slightly better, as it has a smaller overhead (not using carrier-packets). The performance of ICTP becomes better as soon as packet loss is introduced.

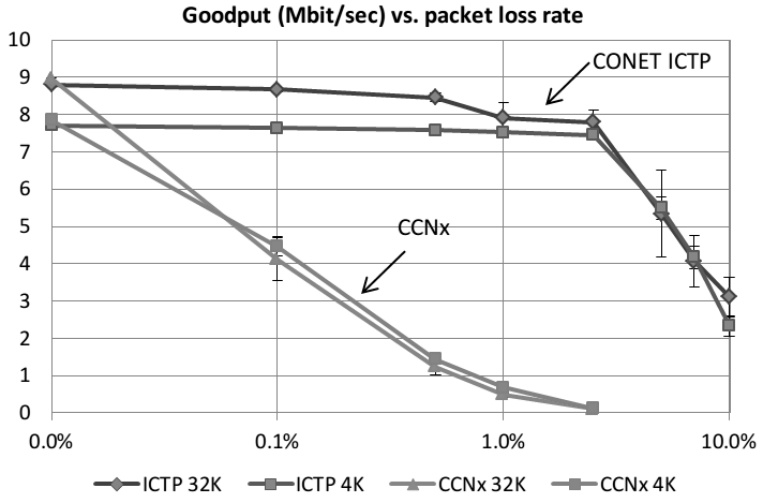

Figure 15: CONET vs. CCNx transport of named-data

3.11 Conclusions

Our aim in designing the CONET was to achieve the promised advantages of ICN (listed in Chapter I), while mitigating its two main cons: 1) ICN requires changes in the basic network operation, which per se is already a big obstacle to take-up of this approach; 2) ICN raises scalability concerns, as the number of different contents and corresponding names is much bigger than the number of host addresses; this has obvious implications on the size of routing tables and on the complexity of lookup functions; in addition, in some proposed CCN architectures (V. Jacobson, et al 2009), providing reverse paths (for information delivery) requires maintaining states in network nodes.

As for the first point, we looked for graceful incremental solutions, backward compatible with the current Internet, as opposed to risky clean slate and flag-day solutions. As regards the second point, we proposed some specific solutions mitigating this problem.

More in details: our routing-by-name architecture is able to support current and future Internet traffic with off-the-shelf technologies; our transport mechanism has performance close to those of the current TCP; our API is similar to that of HTTP facilitating the development of HTTP-to-CONET proxy for the support of legacy applications; our push-based services allow supporting legacy interactive services. Thus, we believe that our proposed CONET can allow an advantageous and smooth migration from the current Internet based on the TCP/IP API, towards a future Internet based on the ICN API.

References

A. Detti, N. Blefari Melazzi, S. Salsano, M. Pomposini, “CONET: A Content Centric Inter-Networking Architecture”, ACM SIGCOMM Workshop on Information-Centric Networking (ICN 2011), August 19, 2011, Toronto, Canada

A. Detti, S. Salsano, N. BlefariMelazzi, “IPv4 and IPv6 Options to support Information Centric Networking”, Internet Draft, draft-detti-conet-ip-option-02, Work in progress, October 2011b

A. Detti, M. Pomposini, N. Blefari Melazzi, S. Salsano, “Supporting the Web with an Information Centric Network that Routes by Name”, Elsevier Computer Networks, vol. 56, issue 17, 2012, p. 3705–3722

A. Detti, A. Caponi, N. Blefari-Melazzi, ” Exploitation of Information Centric Networking Principles in Satellite Networks”, IEEE ESTEL 2012, Roma, Italy, 2-5 October 2012b

A. Detti, M. Pomposini, N. Blefari Melazzi, S. Salsano, A. Bragagnini, “Offloading cellular networks with Information-Centric Networking: the case of video streaming”, IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2012c

A. Detti, A. Caponi, G. Tropea, N. Blefari-Melazzi, G. Bianchi, “On the Interplay among Naming, Content Integrity and Caching in Information Centric Networks”, submitted for publication 2013, available at http://netgroup.uniroma2.it/Andrea_Detti/papers/conferences/ICN-Naming-Signature-Caching-Interplay.pdf

A. Ghodsi, T. Koponen, B. Raghavan, S. Shenker, A. Singla, and J. Wilcox , "Information-Centric Networking: Seeing the Forest for the Trees", in Proc. of the 10th ACM Workshop on Hot Topics in Networks (HotNets-X), November 14-15, 2011, Cambridge, MACambridge, Massachusetts.

A. Kuzmanovic, E.W. Knightly. “Receiver-Centric Congestion Control with a Misbehaving Receiver: Vulnerabilities and End-point Solutions”, Elsevier Computer Networks. 2007, 51, 2717–2737

CCNx project web site: http://www.ccnx.org

D. Cheriton, M. Gritter, “TRIAD: a scalable deployable NAT-based internet architecture”, Technical Report (2000)

D. Perino, M. Varvello, “A Reality Check for Content Centric Networking”, ACM SIGCOMM Workshop on Information-Centric Networking (ICN 2011), August 19, 2011, Toronto, Canada

D. Smetters, V. Jacobson: “Securing Network Content”, PARC technical report, October 2009

D. Trossen, M. Sarela, and K. Sollins: "Arguments for an information-centric internetworking architecture" SIGCOMM Computer Communication Review, vol. 40, pp. 26-33, 2010

K. Changhoon, M. Caesar, A. Gerber, and J. Rexford. "Revisiting route caching: The world should be flat." Passive and Active Network Measurement (2009): 3-12.

L. Chiariglione, A. Difino, N. Blefari Melazzi, S. Salsano, A. Detti, G. Tropea, A. C. G. Anadiotis, A. S. Mousas, I. S. Venieris, C. Z. Patrikakis: “Publish/Subscribe over Information Centric Networks: a Standardized Approach in CONVERGENCE”, Future Network & Mobile Summit 2012, 4 - 6 July 2012, Berlin, Germany

L. Galluccio, G. Morabito, S. Palazzo, “Caching in information-centric satellite networks”, IEEE ICC 2012, June 2012, Ottawa, Canada

L. Popa, A. Ghodsi, and I. Stoica. 2010 “HTTP as the narrow waist of the future internet”, in Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks (Hot-nets-IX). ACM, New York, NY, USA

N. Blefari-Melazzi, A. Detti, G. Morabito, S. Salsano, L. Veltri, ” Supporting Information-Centric Functionality in Software Defined Networks”, IEEE International Conference on Communications (ICC 2012)

N. Blefari Melazzi, A. Detti, M. Pomposini, S. Salsano: “Route discovery and caching: a way to improve the scalability of Information-Centric Networking”, IEEE Global Communications Conference 2012, Anaheim, California, Dec., 3-7 2012b

N. Blefari Melazzi, A. Detti, M. Pomposini, “Scalability Measurements in an Information-Centric Network”, Measurement-based experimental research: methodology, experiments and tools”, Springer Lecture Notes in Computer Science (LNCS), vol. 7586, 2012c

N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “OpenFlow: Enabling Innovation in Campus Networks”. White paper. March 2008 (available at: http://www.openflow.org).

Patrick Th. Eugster, Pascal A. Felber, Rachid Guerraoui, and Anne-Marie Kermarrec. “The many faces of publish/subscribe”, ACM Comput. Surv. 35, 2 (June 2003), 114-131.

R. Braynard, “VoCCN: voice over content-centric networks”, Proceedings of the 2009 Workshop on Re-architecting the Internet (ReArch 2009); 2009 December 1; Rome, Italy. NY: ACM; 2009; 1-6.

S. Salsano, A. Detti, M. Cancellieri, M. Pomposini, N. Blefari Melazzi, “Transport-layer issues in Information Centric Networks”, ACM SIGCOMM Workshop on Information-Centric Networking (ICN 2012), August 17, 2012, Helsinki, Finland

T. Koponen, M. Chawla, B.G. Chun, A. Ermolinskiy, Kye Hyun Kim, S. Shenker, I. Stoica: “A data-oriented (and beyond) network architecture”, Proc. of ACM SIGCOMM 2007, August 27-31, Kyoto, Japan

V. Jacobson, et al., ”Networking named content”, in Proc. of ACM CoNEXT 2009, December 1-4. Rome, Italy

V Jacobson, et al. “VoCCN: voice over content-centric networks”, Proceedings of the 2009 Workshop on Re-architecting the Internet (ReArch 2009); 2009 December 1-4; Rome, Italy. NY: ACM; 2009b.

X. Zhao, D. J. Pacella, and J. Schiller, “Routing Scalability: An Operator’s View”, IEEE Journal on Selected Areas in communications, vol. 28, no. 8, October 2010