CONET: A Content Centric Inter-Networking Architecture

A. Detti, N. Blefari-Melazzi, S. Salsano, M. Pomposini
University of Rome "Tor Vergata"
Via del Politecnico 1, Rome (Italy)

{andrea.detti, blefari, stefano.salsano, matteo.pomposini}@uniroma2.it

ABSTRACT

Content Centric Networking (CCN) is a new networking paradigm in which the network layer provides users with contents, instead of providing communication channels between hosts, and is aware of such contents. In this paper, we introduce a CCN architecture called CONET. A CONET is an *inter-network* that provides users with a network access to remote named-resources (e.g. contents). CONET interconnects CONET SubNets, which can be layer-2 networks (e.g. Ethernet), layer-3 networks, e.g. IPv4 or IPv6, or overlay links (e.g. UDP tunnels over IP). CONET is able to support the already proposed "clean-slate" and "overlay" approaches to CCN. In addition, we propose an "integration" approach, which extends the IP layer with a new header option that supports CCN functionality. CONET limits the size of name-based routing tables by including only a subset of all named-resources; missing entries are looked up in a *name-system* (DNS like) and then cached. CONET does not maintain states in network nodes. We also present two performance studies on the proposed routing approach and on the proposed IP option. [1]

Keywords

content-centric networking, route-by-name, stateless protocols, scalability, in-network caching, route caching, hybrid routing table, IP option.

1. INTRODUCTION

Several papers (see [1][2][3][4] and references therein) and research projects ([5][6]) propose a shift from "host-centric networking" to "information centric" or "content-centric" networking. The essence of Content-Centric Networking (CCN; we will speak of CCN in the following to denote the general trend) is that the network layer provides users with contents, instead of providing communication channels between hosts and is aware of such contents, at least in the sense of knowledge of the name of the contents. A CCN architecture should:

- address contents, adopting an addressing scheme based on names, which do not include references to their location;
- route a user request, which includes a "destination" content-name, toward the closest copy of the content with such a name (name-based anycast routing) and deliver the content to the requesting host;
- provide a native, in-network caching functionality to achieve efficient content delivery both in fixed and mobile environments [8];
- exploit security information embedded in the content to avoid the diffusion of fake versions of contents and to protect the content, as opposed to exploit connection-based or application-based security; protecting information at the source is more flexible and robust than delegating this function to applications, or securing only the communications channels [4];
- provide a way to differentiate the quality perceived by different services [9].

Among the advantages of CCN, extensively discussed e.g. in [3][4], in this paper we want to focus on the improved and built-in support of a replication/caching functionality. Users can "retrieve desired content regardless of where it comes from – the original source, a copy on their local disk, or the user next to them in Starbucks™" [4]. It is true that content replication is already supported by CDN systems [10], but CDNs are private and closed facilities. It is also true that in-network caching is already supported by so called-transparent proxy technologies, but this is done at application level and requires a stateful tracking of user connections. Of course, stateful procedures limit the application of caching in high-speed nodes, where it would be more useful, and where a stateless CCN could instead recognize and cache contents on the fly.

On the cons side, CCN has some drawbacks and challenges. A first, obvious, con is that it requires changes in the basic network operation, which per se is already a big obstacle to take-up of this approach.

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 obvious implications on the size of routing tables and on the complexity of lookup functions; ii) in some proposed CCN architectures [3], guaranteeing bidirectional communication (reverse paths) requires maintaining states in network nodes. This very argument was maybe too heavily used against the Integrated Services architecture (and the RSVP protocol) but it is surely an issue that deserves careful investigations.

When CCN is meant as a replacement of the current network layer, it poses the challenge of how to efficiently support communication sessions based on models different from content retrieval (e.g. http connections for e-commerce applications, instant messaging, social

[1] The CONET architecture has been devised in the framework of the CONVERGENCE project [7], which aims at enhancing the Internet with a content-centric, publish-subscribe service model, based on a common container for any kind of digital data, including representations of people and Real World Objects.

networking; rtp connections for real-time communications). These communication sessions rely on host addresses, and need suitable solutions to work in a CCN environment, as shown in [11] for SIP based VoIP applications.

The goal of this paper is to introduce a CCN architecture that tries to achieve the pros of CCN, and specifically a built-in caching functionality, while mitigating the cons.

Our CONET is an (inter-)network layer that provides users with a network access to remote *named-resources*, rather than to remote hosts. A named-resource can be a data (*named-data*) or a service-access-point (*named-sap*) identified by a network-identifier (NID), that is a name. By default, the NID is an anycast address and CONET may contain multiple replicas of the same named-resource.

In the case of a named-data (e.g. a file), CONET enables users to retrieve it by using the best set of networked devices (caches, mirror servers, etc.) that can provide the named-data. In the case of a named-sap (e.g. the logical port of a server), CONET provides the means to exchange point-to-point data between a requesting entity and an entity addressed by such named-sap. The named-sap case can be extended to multicast, so that CONET provides also the means to exchange point-to-multipoint data between a requesting entity and a set of entities addressed by the same named-sap (in this case the NID of a named-sap has a multicast meaning, rather than an anycast one).

CONET interconnects underlying sub-networks similarly to what IP does. Sub-networks can be layer-2 networks, e.g. Ethernet, or layer-3 networks, e.g. IPv4 or IPv6. Before presenting the details of our solution we summarize its main features:

- is stateless: it does not require maintaining states in network nodes;
- limits the size of name-based routing tables by including only a subset of all named-resources (e.g. the most popular ones); missing entries are looked up in a *name-system* (DNS like) and then cached;
- supports the already proposed "clean-slate" and "overlay" approaches to CCN. In addition, we propose an "integration" approach, which extends the IP layer with a new header option providing CCN functionality [12];
- supports a forwarding table that can logically combine a traditional IP Forwarding Information Base (FIB) and a name-based table (using NIDs). The forwarding table contains also the NIDs of contents stored in the local cache, to support fast cache lookup. Thus, the table can be an hybrid one, in the sense that contains both traditional IP network addresses and NIDs.

In the following we will substantiate these statements. For lack of space, we do not deal with security issues and we only consider the transfer of named-data, neglecting the use of CONET for named-sap (D3.1 in [7] provides further details on the use of named-sap).

2. The content inter-network (CONET)

2.1 Network Architecture

The CONET is composed of a set of *CONET nodes* interconnected by *CONET SubNets* (CSN). CONET nodes exchange *CONET Information Units (CIUs)*, which are used to convey both requests of named-resources, called *interest CIUs*, and chunks of named-resources themselves (i.e., part of files, videos, etc.), called *named-data CIUs*.

A CONET SubNet interconnects two or more CONET nodes, providing transfer of CIUs by using an *under-CONET* technology. Under-CONET technologies can be: point-to-point Layer 2 links, Layer 2 networks, overlay links (e.g. UDP over IP) or arbitrarily large IP networks.

To best fit the transfer units of an under-CONET technology, both interest-CIUs and named-data CIUs are carried in small CONET data units named *carrier-packets*, which also include routing information of the end-to-end communication session they serve to.

The different types of CONET nodes are shown in Fig. 1. *End-nodes* are user devices that request named-resources by issuing interest CIUs. *Serving-nodes* store, advertize and provide named-resources by splitting the related sequence of bytes in one or more named-data CIUs. *Border-Nodes* forward carrier-packets by using CONET routing mechanisms, may reassemble carrier-packets and cache the related named-data CIU, and may send back cached named-data CIUs.

When the CSN is an IP network (IP-CSN), the internal devices use an autonomous IP addressing-space and an interior gateway protocol to set up IP reachability among them. For instance, an IP-CSN could be the IPv4 network of a novel Internet Service Provider that uses a private addressing space and IS-IS intra-domain routing protocol [13]. Within an IP-CSN, optional *Name System Nodes* may be used to assist the CONET routing operations (see Sec. 3). Moreover, optional CONET *Internal-Nodes* could be deployed inside an IP-CSN to provide additional in-network caches. We observe that, albeit optional, internal-nodes are the only ones that could provide caching functionality, when the end-node and the serving-node belong to the same IP-CSN; indeed no border-node is traversed in this cases.

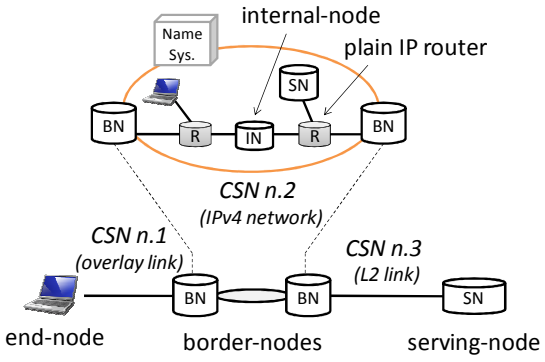

Fig. 1 - CONET Architecture

This network architecture allows deploying the CONET technology by means of three approaches:

- *overlay approach*: CONET on top of an IP network; CSNs are overlay links, e.g. UDP/IP tunnels, as it occurs between end-node and border-node in Fig. 1;
- *clean slate approach*: CONET on top of layer-2 technologies (e.g. Ethernet, PPP, MPLS LSP); CSNs are layer-2 links, and CONET replaces the IP layer, as it occurs between border-node and serving-node in Fig. 1;
- *integration approach*: CONET functionality are integrated in the IP protocol by means of a novel IPv4 option [12] or by means of an IPv6 extension header.

While different variants of the clean-slate and overlay approaches have been already discussed in the literature and by research projects [3], [5], [6], the proposed integration approach is novel to the best of our knowledge; therefore in this paper we focus on this approach, describing it in Sec. 4. We also note that within our proposed architecture, the three approaches are not mutually exclusive, but they can be combined.

2.2 Model of operations

This section provides an example of CONET operation in the scenario depicted in Fig. 1, considering an end-node that retrieves a chunk of named-resource from a serving-node. For the sake of clarity, we assume that the named-data CIU that contains the chunk is fully contained in a single carrier-packet. We also assume that the CONET routing information that enables to *reach-by-name* the named-resource has been already distributed in the CONET. This process is initiated by the serving-node that advertises the named-resources by using a name-based routing protocol, as described in Sec. 3.

The operations performed to provide a end-node with a chunk of named-resource can be described as follows:

- an end-node requests a chunk of a named-resource by issuing an interest CIU, which includes the network-identifier (i.e., the name of the resource); the interest CIU is encapsulated in a carrier-packet, named *I*.
- name-based routing engines in the end-node and intermediate border-nodes route-by-name the packet *I* upward the proper serving-node. Route-by-name means that, on the base of the network-identifier contained in the carrier-packet *I*, a name-based routing-engine singles out the *CSN address* of the next upward border-node toward the serving-node. A CSN address is an interface address, consistent with the CSN technology (e.g., an IPv4 address in case of an IP-CSN). Then, the name-based routing engine encapsulates the carrier-packet *I* in a data-unit of the underlying CSN technology and uses the CSN address as the destination address;
- the CSN address of the end-node and the set of CSN addresses of the traversed border-nodes in the "upward" path toward the serving-node are appended to the carrier-packet *I*, within a control field named *path-state*;
- if present, the internal-nodes forward the carrier-packet *I* by using the underlying routing engine (e.g. IP RIB), but are able to parse the carrier-packet *I*;
- the first in-path CONET node, border, internal or serving-node, which is able to provide the chunk of the named-data requested by *I*, will send back the named-data CIU, without further propagating the interest CIU;
- the named-data CIU is encapsulated in a carrier-packet, named *C*. The carrier-packet *C* follows the same path of the carrier-packet *I*, but in the downward direction and will reach the requesting end-node;
- the reverse-path routing is carried out in a source-routing fashion by using a path-state control information appended to the carrier-packet *C*. This path-state is the copy of the one set up in the interest CIU during the upward routing;
- all the border-nodes and internal-nodes in the downward path may cache the named-data CIU, according to their policies and available resources.

We observe that, differently from [3], we convey the path-state in the carrier-packet; in other words, we include the state of the reverse-path within the packets, rather than having this "pending" state in the traversed nodes. We also observe that in the case of end-to-end sessions bounded within the same IP-CSN, the path-state overhead is not necessary, as it would be composed only of the IP address of the end-node, already contained in the IP header.

2.3 CONET protocol stack

Fig. 2 reports the protocol stack. In every CONET node we can find the CONET and the *Under-CONET* layers. The CONET layer is connectionless, it handles CONET CIUs and carriers-packets, and provides other functionality (e.g. caching, security check, etc.).

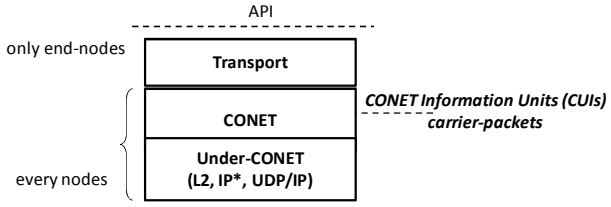

Fig. 2– Protocol stack

The end-nodes has also transport-level functionality, supporting reliability and flow control, and providing the application programming interface (API), see D3.1 [7] for the definition of the API between CONET and upper layers. We adopt the receiver-driven TCP-like approach proposed in [3], which we briefly recall in the following, adapting it to our terminology. The transport algorithm issues a sequence of interest CIUs and each of them requests only a small part of a whole named-resources, e.g. 1500 bytes per interest CIU. By controlling the sending rate of these interest CIUs, it is possible to obtain a TCP-like flow control mechanism. For instance, we could replace current TCP ACKs with interest CIUs and apply TCP congestion-window concepts to in-fly interest CIUs.

Fig. 3 shows the packetization process and the CONET CIUs (interest and named-data) and carrier-packets.
We started from the names and structures proposed in [3] and introduced some modifications both in notation and in functionality. As for the notation, the "interest packets" and "data packets" proposed in [3] correspond to our interest CIU and named-data CIU, respectively, but their protocol information is different (e.g. segment info). In addition, we introduce the concept of carrier-packets, with the goal of improving the forwarding speed of the CONET.

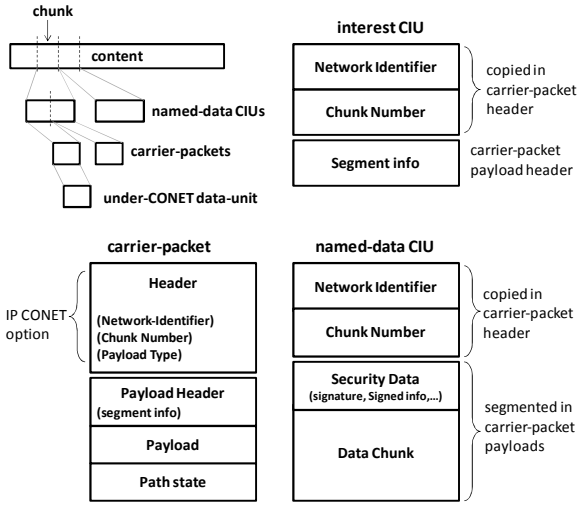

Fig. 3 –Content Information Units (CIUs) and carrier-packets

A named-resource is split in different chunks. Each chunk is inserted in a named-data CIU, whose control information contains the network-identifiers, the chunk number and the security-data [4]. The security-data makes it possible to validate a named-data CIU before caching it or (in the end-node) delivering it to the API. The optimal chunk size comes out from several tradeoffs; we argue that the chunk size should be in the order of the one currently used in P2P systems, e.g. 256-512 kB, nevertheless the CONET architecture could support variable chunk sizes.
The network-identifier is a 3-tuple <*namespace*, hash(*principal*), hash(*label*)>. The network-identifier must be unique in the whole CONET; i.e. different named-resources must use different network-identifiers. We use the concept of principal and label proposed in [2]; furthermore, to support naming heterogeneity we add the possibility of having different namespaces. Principal and label are flat-names and an hash function transforms them to a fixed number of bytes (e.g., 6 bytes). A principal is the owner of her named-resources and uses a principal identifier whose hash is unique in its namespace. Label is an identifier chosen by the principal to uniquely differentiate her named-resources. Each namespace follows its governance rules to release unique principal identifiers. For instance, a namespace named "www" could support the actual WEB names, principals identifiers could be DNS domains, like cnn.com, and to obtain a principal identifier the principal could interact with a naming authority.

An interest CIU is a request of a set of bytes of a chunk of a named-resource, e.g., from byte 1000 to byte 2000 (segment info field) of the chunk n. 3 (chunk number field).
Carrier-packets are low-level carriers of CIUs. Carrier-packets are the data-units of the forwarding process. Carrier-packets includes the path-state control information strictly related to the specific communication session between an end-node and a serving-node (or a cache). Carrier-packets are reassembled in border-nodes, in internal-nodes which want to cache the related named-data CIU, and in end-nodes; this operation is necessary to validate the content.
We introduce carrier-packets because a named-data CIU could be too large to be transported by a single under-CONET data-unit and thus needs to be segmented. A named data CIU can correspond to a chunk of 256/512 kbytes, while the maximum data unit size of under-CONET technologies can be smaller (e.g. 1500 bytes for Ethernet, 64 kbytes for IP).
As shown in Fig. 3, a carrier-packet has the following structure: i) a *header* field, which transports a minimal set of control information of the CIUs, i.e. network-identifier, chunk number and a field to indicate the CIU type (e.g. interest or named-data); ii) a *payload-header*, which identifies the byte boundaries of the carried segment (segment info); iii) the *payload* (existing only in the case of named-data CIU), it contains a part of the sequence of bytes contained in the security-data and data-chunk fields of a named-data CIU; iv) the *path-state* field, previously described in 2.2.

3. Name-based routing: lookup-and-cache

The name-based routing is the mechanism used to update CONET name-based routing tables, which are used by end-nodes or border-nodes to route-by-name interest CIUs. An entry of the name-based routing table contains the tuple <network-identifier, mask, next-hop, output-interface>; it is like an IP routing table entry, but instead of net-prefixes we have *name-prefixes*, i.e. couples <network-identifier, mask>. Next-hop is the CSN address of the next border-node toward the serving-node, as outlined in Sec. 2.2.
In [1][3] the authors suggest to use traditional routing protocols, e.g. BGP or OSPF, to disseminate name-prefixes. We name these approaches *prefix-dissemination*.
We argue that prefix-dissemination could produce big name-based routing tables, because the aggregation of names (i.e., network-identifiers) is not effective when names do not include information about "where" is the serving node [14]. For instance, if we want to support DNS domain names (as we do), a possible location-based aggregation could be done on the basis of top level domains [3]. However, in the case of generic top level domains (.com, .net, etc.) this would not be effective, as current names are geographically very spread (and numerous: .com names are currently about 90 millions). We performed a measure also on the .it country-code top level domain and

found out that about 30% of .it names are outside Italy, which means that the aggregation would not be very effective also in this case.

To support the cases in which name-prefix aggregation is not effective and since it is does not seem feasible to include all names in the routing table, we propose a novel name-based routing, which we name *lookup-and-cache*. In this approach, a CONET node (end-node or border-node) uses a fixed number of rows of its name-based routing table as a *route cache*. When a node misses the routing info required to route-by-name an interest CIU, it looks up its routing entry in a *name-system* (DNS like) and inserts this entry in the route cache. When all rows are filled in, new routing entries may substitute old ones according to suitable policy. From a logical point of view, a name-system serves a single CSN and a specific namespace.

If a serving-node is inside a CSN, the name-system returns the CSN-address of the serving-node. If the serving-node is outside a CSN, the name-system returns the CSN-address of the egress border-node. If there are more than one serving-node, or egress border-node (due to replication operations), the name-system selects the most convenient destination (e.g. according to known techniques [10]).

Prefix-dissemination and lookup-and-cache approaches can work separately or they can be combined, e.g. by using prefix-dissemination for the most popular named-resources and lookup-and-cache for the remaining ones.

4. Integrating CONET in IP

In this section, we describe a technique to support the CONET in a CSN that is an IP network, e.g. the IPv4 network in Fig. 1. The IP network can correspond to the whole public Internet; therefore this technique is a way to offer CONET services in the Internet. We propose a so-called *integration approach*, which: i) does not imply to give up IP, as in the *clean-state approach*; ii) performs better than a CONET placed on top of IP, as in the *overlay approach*. The idea of the integration approach is to make IP itself content-aware, as follows. We transport the header of a CONET carrier-packet in a novel IPv4 option (or IPv6 extension header), which we name CONET option (see Fig. 3 and [12]). Border and internal CONET nodes of an IP-CSN are nothing else than IP routers extended with CONET functionality.

Fig. 4 shows a possible architecture of a border or internal CONET node. We have a *fast forwarding path* that handles forwarding operations for CONET carrier-packets and for plain IP packets. The hardware (RIB or FIB) routing table does not only include IP net-prefixes but also name-prefixes, which address both remote named-resources and local cached named-resources. The latter entries point to the local cache engine. Other CONET and IP functions with less stringent delay constraints are performed by a CPU. For instance, the CPU performs IP and name-based routing, implements caching algorithms, reassembles named-data CIU to cache them, replies to interest CIUs that request a cached named-resources, etc. Most of these operations require to parse incoming CONET CIUs, which are "copied" in the CPU while at the same time being forwarded by the HW engine.

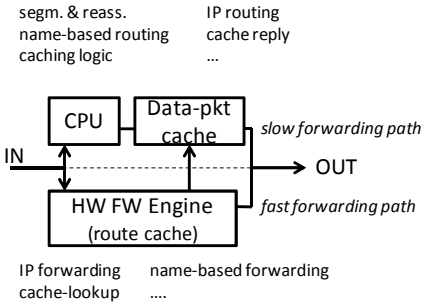

Fig. 4 – Architecture of a CONET node of an IP-CSN

The advantages of this approach with respect to the overlay one is that it allows CONET nodes to quickly forward carrier-packets, without the need of a slow deep packet inspection. This is a fundamental requirement to deploy content-centric features in backbone nodes, where the high packet rates demand a fast forwarding operation. In addition, this approach allows deploying CONET routing-by-name functions only in a subset of nodes (i.e. border-nodes and end-nodes) while allowing to perform caching in all nodes running the new IP option (i.e. internal nodes). On the contrary, in the overlay approach, caching in all nodes would require to deploy routing-by-name functions functionality in all nodes.

The disadvantage of the integration approach is that we require to introduce the new IP option, but this is much less disruptive than the clean state approach.

The integration approach lends itself to different deployment scenarios.

It is possible to think to an extreme case in which an IP-CSN corresponds to the whole Internet and routing-by-name functions are performed only in end-nodes. In-network caching would still be possible simply by introducing the new IP option and without the need of introducing routing-by-name functions within the routers.

Another scenario is to partition the Internet in a set of IPv4 CSNs that interoperate only by using the CONET protocols. Each CSN uses an IPv4 addressing that is unique only inside the CSN and bounds the scope of IP routing to that CSN. Therefore new providers offering public CONET services can be added with no need of public IP addresses, and with no further increase of the current routing table size of Internet backbone routers. This scenario can be a solution to the problem of growing size of Internet backbone routing table [14], shifting the problem into the scalability of the CONET routing-by-name mechanisms, which in any case needs to be addressed in CCNs.

5. Performance checks

This section describes two experimental performance checks carried out to assess the feasibility of some of our

design choices. We focus our attention on the two main novelties of the paper: lookup-and-cache and IP-CONET integration. We refer to a CONET composed by all IP-CSNs, which use the CONET IP option and the lookup-and-cache routing approach. For the experiments, we used real Internet traces [15], [16] and the PlanetLab facility.

5.1 Lookup-and-cache

We remind that name-based routing involves only interest-packets, since data-packets are routed back to the end-node by means of the path-state information (see Section 2.2). The CONET nodes involved in name-based routing are either end-nodes or border-nodes. In case of end-nodes, the lookup-and-cache approach resembles the interaction between an Internet host and a DNS server, where the host implements a local DNS cache service. Therefore, we argue that lookup-and-cache is feasible on end-nodes and we focus on its feasibility in border-nodes.

We assume to replace a standard TCP session between a client and a WEB server with a CONET session (exchange of CONET CIUs) between an end-node and a serving-node, or an intermediate cache. Specifically, we assume that:

- an URL <http://domain/path> is replaced by the network-identifier:
  <namespace="www", principal=domain, label=path>;
- TCP segments are replaced by carrier-packets that convey segments of named-data CIUs;
- TCP ACKs are replaced by carrier-packets that convey interest CIUs (see Sec. 2.3);
- the TCP/IP destination address contained in the TCP ACKs, i.e. <IP address:80>, is replaced by the couple <namespace="www", hash(principal=domain)>
  contained in the header of the interest CIUs. Such a couple is the input of the route-by-name process applied to interest CIUs (see Sec. 3).

With these assumptions, we can map a real Internet trace, formed by TCP segments and ACKs, to a "CONET trace", formed by carrier-packets that convey CONET segments of named-data CIUs and interest CIUs. We applied this re-mapping to two Internet traces: the first one captured on an interface at 10 Gbit/s of a tier-1 router [16]; the second one captured on an interface at 10 Mbit/s of a tier-3 router [15]. The two remapped traces have been fed to a CONET border-node, which we emulated in SW, to analyze the effectiveness of the lookup-and-cache routing for a tier-1 and a tier-3 border-node. Following the approach suggested in [2], we assume here that name-based routing is performed only on the base of the principal identifier. This means that the network-identifier of a name-based routing entry has the form <namespace, principal,*> and that all the named-resources of a given principal are stored in a serving-node (and in its replicas, if any). We also assume that the route cache adopts a Least Recently Used caching policy, discarding the least recently used item first.

Fig. 5 shows the obtained results in terms of name-lookups per second issued by the border-node to the name-system, versus the size of the route cache. We notice that, as in the case of WEB caching, the route caching performance improves (i.e. lower lookup frequency) in a log-like fashion [17] versus the cache size. In the case of the tier-3 node, we have about 2 lookups per second and a cache-miss probability of about 10^{-3}, by using a route cache of 2k entries. In the case of the tier-1 node, we have about 10 lookups per second and a cache-miss probability of about 10^{-4}, by using a route cache of 8k entries.

Considering that nowadays BGP routers handle 350k entries and 2 or 10 lookups per seconds are reasonable values, we conclude that lookup-and-cache seems feasible with the current technology.

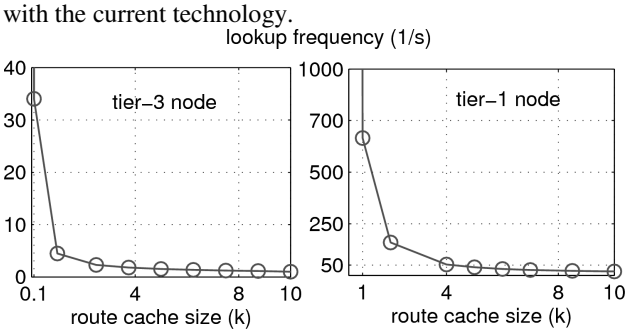

Fig. 5 – Lookup frequency of a tier-3 and a tier-1 border-node

5.2 CONET-IP integration

In this section, we verify the feasibility of conveying the header of carrier-packets in an IPv4 option, i.e. the CONET option. The rationale of this test lies in the fact that IP router manufacturers tend to process packets with IP options in the slow forwarding path; therefore, current IP routers could become a critical performance bottleneck for our solution, as plain IP routes and CONET nodes would need to co-exist in a hypothetical real deployment scenario.

To check the behavior of current IP routers, we sent IP packets with and without our CONET option on the current Internet and we measured the difference in terms of round-trip-delay and throughput (i.e. the available capacity between a sender and a receiver). We used eleven PlanetLab nodes, spread over the Internet (Asia, Europe, North America, Australia). Each measurement was performed between a PlanetLab node and a node in our premises (Rome, Italy).

The measurement of the round-trip-delay and of the throughput are carried out by two C++ tools, which we implemented by using raw sockets. To make a fair comparison, we performed the measurements by simultaneously running two instances of the same tool: an instance issuing packets with the CONET option and the other one without the option; therefore the two packet streams experience similar Internet condition. Each measurement has been repeated ten times and we report the average values in Fig. 6.

As regards the throughput, we observe that we have almost the same performance, with and without the CONET option, for the first nine PlanetLab end-nodes; thus, the worsening due to the router processing on these Internet

paths is not significant. On the other hand, we observed considerable differences in the case of the last two end-nodes. Further analysis of the latter two paths revealed that: i) on the Beijing-Rome path there is a router that implements the policy of statistically dropping half of the packets with IP options; ii) on the Colgate-Rome path there is a router (in Australia) that implements the policy of dropping all packets with IP options. The problem regards a minority of the examined routers, depends on a software configuration and we conjecture that these policies are enforced to prevent DoS attacks [18]; therefore such policies could be easily modified, so as to accept CONET carrier packets without restrictions.

As regards the round-trip-delay, we observe a small increase of the latency for packets with the CONET option. However, the impact seems to be quite limited.

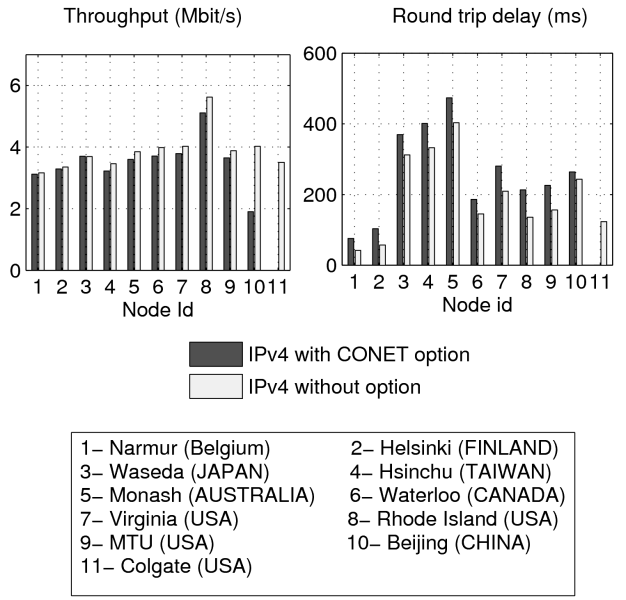

Fig. 6 – Throughput and round-trip-delay of IP packets with and without CONET options on different Internet paths

Overall, the measurement campaign shows that legacy IP routers (properly configured) would not be a critical performance bottleneck, and therefore that the use of the IP CONET option seems feasible, from this point of view.

6. CONCLUSIONS

As a conclusion, let us re-consider, in light of our work, the advantages, the cons and the challenges of CCN, which we discussed in the introduction. We argue that our proposed CONET architecture and technical solutions: i) are able to effectively support in-network caching and content replication; ii) address the issue of changing the network operation by proposing an "integration" approach that can be incrementally deployed in current IP networks; iii) face scalability of name-based routing with the lookup-and-cache approach; iv) do not need to maintain states in network nodes; v) support communication sessions different from content retrieval, either with the support of named-sap (N.B. this was only mentioned in this paper) or thanks to the fact that CONET can smoothly co-exist with IP networks and therefore such communication session could continue to be run on classical IP.

7. REFERENCES

[1] D. Cheriton, M. Gritter, "TRIAD: a scalable deployable NAT-based internet architecture", Technical Report (2000)"

[2] 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

[3] V. Jacobson, et al., "Networking named content", in Proc. of ACM CoNEXT 2009

[4] D. Smetters, V. Jacobson: "Securing Network Content", PARC technical report, October 2009

[5] PURSUIT project website: www.fp7-pursuit.eu

[6] 4WARD project website: www.4ward-project.eu

[7] CONVERGENCE website: www.ict-convergence.eu

[8] K Katsaros, G. Xylomenos, G. C. Polyzos: "MultiCache: An overlay architecture for information-centric networking", Computer Networks, Elsevier, Volume 55, Issue 4, 10 March 2011, Pages 936-947

[9] S. Oueslati, J. Roberts, N. Sbihi: "Ideas on Traffic Management in CCN", Information-Centric Networking, Dagstuhl Seminar, http://drops.dagstuhl.de/opus/volltexte/2011/2943/pdf/dagstuhl_icn_proceedings.2943.pdf

[10] D. C. Verma "Content Distribution Networks", Wiley-Interscience

[11] V. Jacobson, et al "VoCCN: voice-over content-centric networks", Proc. of ReArch '09 workshop, 2009

[12] A. Detti et al., "An IPv4 Option to support Content Networking", Internet Draft, draft-detti-conet-ip-option-00, Work in progress, March 2011.

[13] D. Oran, "OSI IS-IS intra-domain routing protocol", IETF RFC 1142

[14] D. Meyer, L. Zhang, K. Fall, "Report from the IAB Workshop on Routing and Addressing", RFC 4984

[15] Waikato Internet Trace Storage, http://www.wand.net.nz/wits/waikato/1/20050815-000000-0.php

[16] CAIDA Internet Trace Storage, https://data.caida.org/datasets/passive-2010/equinix-sanjose/20101118/

[17] L. Breslau, P. Cao, L. Fan, G. Phillips, S. Shenker, "Web Caching and Zipf-like Distributions: Evidence and Implications", IEEE INFOCOM 1999

[18] F. Gont, S. Fouant, "IP Options Filtering Recommendations", Internet Draft, draft-gont-opsec-ip-options-filtering-00.txt