A BGP-aware discovery service

Luigi Liquori, Rossano Gaeta, Matteo Sereno

A BGP-aware discovery service

Luigi Liquori
Université Côte d'Azur, Inria
Luigi.Liquori@inria.fr

Rossano Gaeta
Università di Torino
Rossano.Gaeta@unito.it

Matteo Sereno
Università di Torino
Matteo.Sereno@unito.it

***Abstract*—Internet in recent years has become a huge set of channels for content distribution. And this has highlighted limits and inefficiencies of the current protocol suite originally designed for host-to-host communication. This paper joins the research efforts addressed by the new Internet challenges by proposing a Content Name System (CNS) discovery service, extending the current TCP/IP hourglass Internet architecture, that provides a new network aware content discovery service. Contents are addressed using "hypernames", whose rich syntax allow to specify hosts, PKI, fingerprint and optional logical attributes (tags) attached to the content name, such as e.g. mutable vs. immutable contents, digital signatures, owner, availability, price, etc.**

The CNS behavior and architecture is, partly, inspired by the Domain Name Service (DNS) Internet service, and whose discovery process logic uses the Border Gateway Protocol (BGP) information allowing Internet to route between different Autonomous Systems (AS).

The service registers and discovers object names in each Autonomous System (AS), and the content discovery process is inspired to the so called "valley-free" property. In the routing among different ASes (i.e., the BGP protocol) this is a property that avoids un-justified AS transit costs.

In our proposal this property allows to implement a network-aware service discovery.

I. Introduction

Information Centric Networks (ICN) is a clean-state approach to redesign the actual Internet infrastructure from a host-centric, fully connected, paradigm to a name-centric, loosely connected, paradigm where the focus is on named data instead of machine name hosting those data. In the last decade many proposals raised from research to capture this new paradigm: they mainly can be grouped into two schools of thought: *(i) Content Centric Networks* referring to the Jacobson-based vision [JST+09], [ZAB+14], [SBL+16], where routing is driven by fully qualified - human readable - hierarchical names and *(ii) Data Oriented Network Architecture* (DONA) referring to a flat, unreadable but unique name-space [KCC+07] (see also [BCA+12], [XVS+14] and the references therein).

Clean-state design vs. evolution of the actual infrastructure is a common dilemma in designing future Internet features: while it is always exciting to conceive a new network starting from new concepts and from a clean-state design, network's history teaches us that the Internet infrastructure and its protocol suite have little changed especially at the lower level of the OSI stack; this is, quite obviously, because of strong backward-compatibility needs, and because of the tremendous expansion of the Internet phenomenon. Therefore, one could not imagine to "switch off" the actual Internet and restart few second later with a totally new protocol suite running on the same network's equipments: these lines of thought claim that it would be reasonable to envisage an IP dismission in the future definition of ICN.

This paper supports the evolutive research line and presents a lightweight network aware Internet service to be implemented between the Transport and the Session layers (referring to the ISO-OSI protocol layering). We call this new service *Content Name System* (CNS) organized throughout a set of communicating CNS servers and the `link` protocol implementing the service discovery. The purpose of this discovery service is to publish machine-IP-addresses being the owners (or the purveyors) of some named-contents and retrieve that machine-IP-addresses performing a distributed search using the named-content as the database-key. The service binds a set of IP-addresses to content-names, called *Hypernames* (`HN`); a hypername is composed by a (possibly) human readable part followed by an optional machine readable part: the content-name is enriched with a number of optional parameters, also called logical attributes or tags, helping to identify univocally contents, ownerships, integrity, signature, price, availability, etc. Attributes are not enforced to be human readable.

In a nutshell, a Content Name System is a hierarchical decentralized discovery naming system translating content-names into IP addresses needed to later retrieve the content itself; all CNS servers are distributed over the Autonomous Systems: more precisely there is at least one CNS per AS (or for load balancing purposes there can be multiple servers) taking care of the content-names registered inside the AS itself. The CNS service stops when some or no IP-addresses are returned or when no other CNS can be delegated in the iterative call implementing the distributed data-base query. Each CNS server is equipped with a database containing for each queried content-name, the set of corresponding IPs ordered by local awareness. The discovery service leverages on AS relationships, or more precisely, the CNS server hierarchy mimics the AS relationship hierarchy. The routing between ASs (also called interdomain routing) is determined by the Border Gateway Protocol (BGP) [RL95]. The main feature of the interdomain routing is that it allows each AS to choose its own administrative policy in selecting the best route, and announcing and accepting routes.

The BGP business relationship between ASs can be classified into two main classes of agreements: *customer to* *provider*, and *peering*[1]. An AS customer pays its providers for connectivity to the rest of the Internet. A pair of ASs can set up a peering relation and in this case they agree to exchange traffic between their respective customers free of charge.

The AS-graph annotated with these two kinds of relationships is one of the most famous and studied representations of the Internet, e.g., see the measurement studies provided by CAIDA [CAI16]. In this graph, according to their roles, we can distinguish three different kind of ASs: Tier-1, Tier-2, and Tier-3. A Tier-1 is an AS that can reach every other destination on the Internet without paying other ASs: in other words, a Tier-1 is an AS with (many) customer ASs but with no provider. For connectivity purpose the Tier-1-s set up peering relationships among them. On the other hand, a Tier-3 is a stub AS, without any transit customers, and with some peering relationships. Tier-3 ASs generally purchase transit Internet connection from Tier-2 ASs and, in some cases, even from the Tier-1 ASs as well. Finally, a Tier-2 is an AS with customers, and some peering, but that still buys transit service from Tier-1 ASs to reach some portion of the Internet.

Note that the relationships among the ASs play a fundamental role in shaping the AS graph structure and in defining the routing policies implemented throughout BGP. In particular, the paths between two ASs must avoid routing policies that would result in unjustified payments by some AS. Examples of such incorrect routing paths are, for instance, an AS provider that routes the traffic directed to another AS provider by forwarding it to one of its AS customers. This path is incorrect because would cause an unjustified cost in charge to the AS customer used as an intermediate. Another example of incorrect path occurs when an AS forwards its traffic by using as intermediate step one of its peering relationships: in this case the peer AS chosen as intermediate would be in charge of the transit cost for the traffic it forwards. Figure 1 presented in [Gao01] shows a simple configuration of seven ASs and their relationships, i.e., provider-to-customer and peering. On this simple AS graph we report two wrong, and two correct paths.

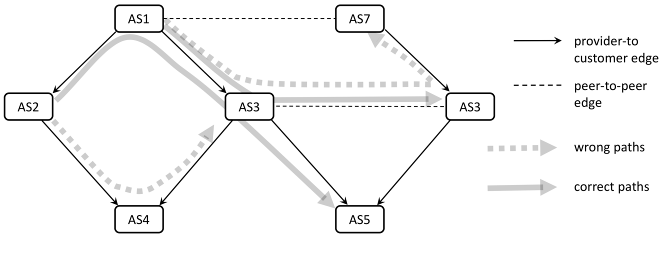

Fig. 1. A simple AS graph with two wrong and two correct routing paths

The routing paths among pairs of ASs is obtained by the BGP protocol that uses selective exporting path rules, i.e., each AS selectively provides transit services for its neighboring ASs. All the paths, also called routes, with property we previously discussed are called *free-valley* or *no-valley* paths [Gao01].

In the CNS service each AS uses the business relationships with the ASs in its neighborhood to drive the content location process according to the no-valley property.

Therefore, the main contributions of the paper are:

- introduction of a discovery service allowing to search contents names through the CAIDA's [CAI16] augmented graph of Autonomous Systems: the service is achieved by defining (i) a hypername naming notation to denote contents, (ii) a distributed Content Name System, installed along the Autonomous System graph, and (ii) the `link` protocol, to route queries along the the AS-graph such that:
    - the `link` protocol respects, by definition, the Gao's [Gao01] abstract business relationship model between provider-customer-peer AS and guarantees that, for every AS providing lookup's transit, there is a payee that is its immediate neighbor in the path, and all the path are valley-free;
    - the hypername expressivity go beyond the Zooko's conjecture [Zoo03], allowing the CNS naming, human readable, decentralized and secure;
- simulations show that `link` is able to successfully locate objects with high probability at low cost; they also show that good performance can be obtained by excluding Tier-1 ASs from the search phase providing that Tier-2 ASs are incentivized to cooperate.

Plan of the paper. In Section II we introduce hypername syntax, the CNS service discovery and the `link` protocol; Section III presents simulations to assess the performance of CNS service. Finally, Section IV discusses some further developments.

II. Content Name System and hypernames

A. Content Name Service

We locate the CNS server in the ISO/OSI hourglass, at the application level, similarly, for instance, to the DNS, CNS translates hypernames into lists of IP addresses, that is $\texttt{HN} \Longrightarrow \{\mathrm{IP_i}\}^{\mathrm{i}\in\mathrm{I}}$ with $\mathrm{I} = \emptyset$ in case of discovery failure. Because of its similarity with DNS service, we introduce the CNS service by putting it face-to-face with DNS.

- DNS [Moc83] is a fundamental phone book directory for the Internet. It mainly uses the UDP transport to query other distributed DNS servers to answer client questions like "which IP addresses are associated with the name `www.google.com`?" The DNS service provides information about hosts querying the DNS hierarchy: this hierarchy can goes through ASs and do not follow the AS cash flow route: the small amount of packets involved in DNS resolution makes DNS economically scalable. On the contrary, and this will be made explicit in the `link` pseudocode, packets will be routed following the economic interest of the AS that generates the query: this

[1] The ASs can establish also other type of relationships such as sibling and backup. For the purposes of this paper we neglect them.

point is crucial for ensuring the discovery service to be economically scalable.
- DNS delegates name resolution into domain zones from the smallest to the biggest zone. With the same idea, the CNS delegates content-name resolution through ASs always trying to follow, when possible, the cash flow route suggested by BGP;
- DNS distributed database is indexed via domain names. On the other hand, the relations among CNS servers are derived by the relations among ASs (customer-to-provider, provider-to-customer and peering relations). These relations can be derived by using CAIDA's AS relationships dataset maps (see [CAI16] and Gao's pioneering work [Gao01] on valley-free);
- DNS queries can be iterative or recursive: the same holds for CNS: nevertheless, an efficient implementation of the CNS service prefers iterative queries.

B. CNS hierarchical topology

The CNS distributed database is organized into a hierarchy of servers distributed according to the classical Tier-1/Tier-2/Tier-3 AS topology as in the introduction.

As well explained by CAIDA, an annotated AS/ISP graph includes relations of kind customer-to-provider, provider-to-customer and peering-peering. The customer-to-provider autonomous system relation refers to a relation where the customer ISP pays the provider ISP for transit (the, so called, "flow of money"). Therefore, autonomous systems at lower levels pay ISPs at higher levels in exchange for access to the rest of the Internet. A peering link, instead, connects two autonomous systems who have agreed to exchange traffic on a quid pro quo basis. Autonomous system involved in a peering relation exchange traffic only between each other and each others customers. Each autonomous system must have at least one CNS, called authoritative, whose database will take into account the association of each hypername with a list of IPs that have registered a content named by a hypername. The authoritative CNS also knows exactly its position in the distributed database, namely (i) the IP addresses of all customer CNSs, (ii) the IP addresses of all provider CNSs and (iii) the IP addresses of all peer-to-peer CNSs: this will allow to dispatch queries along the distributed database.

C. Content publication

In order to make a content-name discoverable, the owner or purveyor published an hypername referring to a content-name in the local CNS. Note that the publication in a CNS associates the hypername with a principal, and that principal holds the content as an owner or a purveyor (the content being mutable or immutable). Suppose a given content C be available by a host belonging to an autonomous system: the host can publish, through the CNS service, the content in the authoritative CNS local database. To do this, at the beginning, the host creates a proper hypername that will be sent as a formal parameter to the authoritative CNS. Note that the host decides which attribute to attach to the hypername and if it should publish that content as a owner or as a purveyor. In the first case (owner) the publication is done by a simple write in the CNS database[2]. In the second case (purveyor) the host could be asked to package a `.torrent` file and write it in the CNS database. Following the bittorrent jargon, the purveyor plays a role of seed and it will be asked to publish itself as a purveyor of the content C every time interval: further nodes entering the swarm for C will be asked to publish his name in the torrent; for that content, the CNS server would serve as a network aware bittorrent tracker.

D. Hypernames (`HN`*)*

A hypername is a human readable string denoting the content-name, enriched with a number of optional parameters to identify its ownership, its integrity, its hosting, and its attribute-list. Hypernames are generated by the following abstract syntax:

```
cont_name[:tag_list][:host_list][:fingers]
```

where

```
tag_list ::= string | string,tag_list
host_list ::= IP | IP,host_list
fingers ::= fing_cont,fing_princ
```

Intuitively:

- `cont_name` is a human readable string denoting a content-name (e.g., "openoffice.iso", "traffic_light", "defibrillator", "plastic_bottle", "tv_display", "wifi_spot", URI, MAC, GUID, etc.);
- `tag_list` is an optional human readable list of keywords (e.g., "sell", "buy", "rent", "cars", "b&b", etc) associated with a given content;
- `host_list` is an optional list of hostnames being the purveyors of the content: when a hypername contains a list of hostnames, then the content-name is retrieved from one of the hostnames: the local CNS perform a DNS query, transforms one (or all) hostname(s) into IP address(es) and return that list to the sender of the discovery request;
- `fing_cont` is an optional digital signature (hash) denoting the integrity of the content to be retrieved;
- `fing_princ` is an optional digital signature denoting the public asymmetric key of the principal, i.e., the owner of the content: it allow to identify the identity of the latter as soon as we retrieve the content itself.

Therefore, a hypername is characterized by an "external" and an "internal" view: the external view only includes a content-name and a tag list. Optional parameters are used to improve efficiency in pattern matching algorithms[3] or to guarantee content's security or the identity of the purveyor. Thus, external and internal views break Zooko's triangle conjecture [Zoo03], [GKR$^+$11] saying a naming cannot achieve more than two of

[2] Depending on a local policy, the CNS could ask to republish the content every n seconds.

[3] Pattern matching issues are out of the scope of the paper.

the three features, namely to be human readable, decentralized, and secure.

E. The `link` *discovery protocol*

Each Autonomous System holds an authoritative CNS server, that records the mappings for all the hypernames published inside it. Hence, the CNS distributed database is organized hierarchically following CAIDA's AS graph augmented with providers-customers-peers relationships [CAI16]. Each AS has three numeric parameters $\alpha, \beta, \gamma \in [0, 1]$ denoting the number of customers, providers and peers to forward `link` packets. Tuning those parameters is crucial in order to optimize success discovery, minimize the cash-flow of the discovery itself and avoid flooding.

Intuitively, the `link` protocol proceeds as follows:

1) an IP client looking for a content denoted by a hypername contacts its authoritative CNS which searches the hypername in its database;
2) if the above fail, then the authoritative CNS forwards the query through the α CNS associated with the ASs in downstream (in Gao jargon "downhill"), i.e., with which we have signed some provider-to-customer agreement;
3) if the above fails, then the authoritative CNS forward the query through the γ CNS associated with the ASs in peer, i.e., with which we have signed some peering agreement;
4) if all of the the above fail, then the authoritative CNS forward the query through the β CNS associated with the ASs in upstream (in Gao jargon "uphill"), i.e., with which we have signed some customer-to-provider agreement.

The `link` pseudocode is presented in Figures 2, 3, and 4. A client sends a query to the local authoritative CNS server, with argument the hypername `HN` and a direction `UP` (from customer-to-provider or from peer-to-peer) or `DOWN` (from provider-to-customer). This query is recursive and the client will be blocked until the CNS will answer positively with a result containing a set of addresses $\{\mathrm{IP_i}\}^{\mathrm{i}\in\mathrm{I}}$ associated with `HN`, or with a search failure.

Location process start. A client sends a query to the authoritative CNS server where the client belongs to, with argument the hypername `HN`. In DNS jargon, this query is recursive i.e., the client will be blocked until the CNS will answer positively with a result containing a set of addresses $\{\mathrm{IP_i}\}^{\mathrm{i}\in\mathrm{I}}$ associated with `HN`, or with a search failure.

Figure 2: from provider with downhill. This code refers to the general case when the current CNS receive a `links` message with a `HN` and a downhill direction from a provider (line 1.01). First of all, a local lookup is performed (1.02); in case of success, the result value is returned to the sender[4] (1.04); else selects α customer (1.05) and sends α-iterative `link` queries with the same `HN` and the same downhill direction (1.06); then collects the result value and send it back to the sender of the first `link` message (1.07). Before return, all the results will be written in the local CNS in order to give a direct answer in successive queries.

Figure 3: from peer with uphill. Following the BGP jargon, this code refers to the case of being "on the top of the hill", i.e., receiving a message from uphill and from a peer. Execute the same code as the one of Figure 2, with the following exception: invert the direction from uphill to downhill when sending α-iterative `link` queries (2.06). Before return, all the results will be written in the local CNS in order to give a direct answer in successive queries.

Figure 4: from customer with uphill. This code refers to the case where a CNS receive a `link` message from uphill from a customer. Following the BGP jargon, when we receive a query from a customer and with an uphill direction the following steps are executed. First of all, a local lookup is performed (line 3.02): in case of success, the result value is returned to the sender (3.04); else select α customer[5] (3.05) and send α-iterative `link` queries with the same `HN` but inverting the direction from uphill to downhill (push downhill the query) (3.06); in case of success, the result value is returned to the sender (3.08); else select γ-peers (3.09) and send γ-iterative `link` queries with the same `HN` and the same direction[6] (3.10); in case of success, the result value is returned to the sender (3.12); else select β providers (3.13) and send β-iterative `link` queries with the same hypername and the same uphill direction (in other words: go uphill only after having tried to invert the search downhill but all the queries failed) (3.14); as the last resort of the query, return a success or failure value to the sender. As in [JST$^+$09], before return all the results will be written in the local CNS in order to give a direct answer in successive queries.

Note. All α, β, and γ are CNS dependent and all messages not matching with the above three figures are flushed by the receiving CNS server.

III. Performance Results

In this section we present results that characterize the performance of the CNS service and of the `link` protocol. Simulations show that the protocol is able to locate popular content at low cost for all autonomous systems involved in the lookup: "low cost" here means that all ASs involved in the queries minimize the overall cost of forwarding packets towards their providers-customers-peers. Simulations also show that good performance can be obtained by excluding Tier-1 ASs from the search phase providing that Tier-2 ASs are incentivized to cooperate[7]; excluding Tier-1 ASs means that providers generating the query will receive results only from customers or peers without any cost (to be precise even earning some money due to the business relationship between the ASs).

[4] At the beginning of the search, the sender is just the authoritative CNS itself, while in the middle of the location process, the sender is a provider.

[5] Do not choose the customer that have sent the query.

[6] Successive execution of code in Figure 3 will later invert the direction from uphill to downhill, i.e., we push downhill the query.

[7] That is, having α and β parameters not so small.

```
on receipt of link(HN,DOWN) from provider do                          receive a query from a ‘‘downhill”
value = lookupdb(HN);                                                 search HN in the CNS' local data base
if (value $\neq$ 0)                                                   some IP publishing HN are found
then {publish(HN,value) to CNS; return value to provider};            write in the local CNS and return IPs ‘‘back to the downhill’’
else list = select($\alpha$,customerlist);                            select some customers CNS
forall cus $\in$ list do value = value $\cup$ send link(HN,DOWN) to cus;   and forward the query downhill through a customer
publish(HN,value) to CNS; return value to provider;                   write in the local CNS and return IPs ‘‘back to the hill’’
```

Fig. 2. `link`: queries from provider with downhill direction continue on α thread downhill

```
on receipt of link(HN,UP) from peer do                                receive a query from a peer on the ‘‘top of the hill”
value = lookupdb(HN);                                                 search HN in the CNS' local data base
if (value $\neq$ 0)                                                   some IP publishing HN are found
then {publish(HN,value) to CNS; return value to peer};                write in the local CNS and return IPs ‘‘back to the top of the hill’’
else list = select($\alpha$,customerlist);                            select some customers CNS
forall cus $\in$ list do value = value $\cup$ send link(HN,DOWN) to cus;   and forward the query but downhill through a customer
publish(HN,value) to CNS; return value to peer;                       write in the local CNS and return IPs ‘‘back to the top of the hill’’
```

Fig. 3. `link`: queries from peer with uphill direction will change on α thread downhill

```
on receipt of link(HN,UP) from customer do                            receive a query from a ‘‘uphill”
value = lookupdb(HN);                                                 search HN in the CNS' local data base
if (value $\neq$ 0)                                                   some IP publishing HN are found
then {publish(HN,value) to CNS; return value to customer};            write in the local CNS and return IPs ‘‘back to the uphill’’
else list = select($\alpha$,customerlist);                            select some customers CNS
forall cus $\in$ list do value = value $\cup$ send link(HN,DOWN) to cus;   and forward the query but downhill through a customer
if (value $\neq$ 0)                                                   some CNS are suggested
then {publish(HN,value) to CNS; return value to customer};            write in the local CNS and return IPs ‘‘back to the uphill’’
else list = select($\gamma$,peerlist);                                select some peers CNS
forall per $\in$ list do value = value $\cup$ send link(HN,UP) to per;     and forward the query uphill through a top of the hill peer
if (value $\neq$ 0)                                                   some CNS are suggested
then {publish(HN,value) to CNS; return value to customer};            write in the local CNS and return IPs ‘‘back to the uphill’’
else list = select($\beta$,providerlist);                             select some provider CNS
forall pro $\in$ list do value = value $\cup$ send link(HN,UP) to pro;     and forward the query uphill through a provider
publish(HN,value) to CNS; return value to customer                    write in the local CNS and return IPs ‘‘back to the uphill’’
```

Fig. 4. `link`: A query from customer with uphill direction will continue on three directions: first α-downhill, then γ-downhill, and finally β-uphill

Performance is represented by two indexes: the *hit probability* (denoted as P_{hit}) that is defined as the fraction of ASs that successfully locate a requested object, and the *average lookup length* (denoted as ALL) representing the average number of CNS servers explored during the search phase. To this end, we developed a C-based emulator of the proposed object discovery service. The emulator runs by using real ASs topologies provided by CAIDA [CAI16] and is able to reproduce the dynamic behavior of location requests. We provide a sensitivity analysis of the lookup algorithm `link` with respect to three numeric parameters $\alpha, \beta, \gamma \in [0, 1]$ denoting the "forwarding" rate of a query in customers, peers, and providers, respectively. We also discuss the performance of `link` as a function of the fraction of ASs that actually deploy a CNS server to support the location service.

A. Scenario

In our experiments we selected an ASs topology provided by CAIDA containing all the ASs and their type of relationships. In these snapshots edges between two nodes either represent peer relationships between ASs (undirected edges) or provider-to-customer roles (directed edges). As said in the introduction, we classify ASs in Tier-1/Tier-2/Tier-3 [CAI16] subsets based on the topological characteristics of nodes and we recall that (i) Tier-1 are noded those snapshot nodes with no incoming edges, i.e., ASs that have no providers (ii) Tier-3 those snapshot nodes with no outgoing edges, i.e., ASs that have no customers and (iii) Tier-2 all other snapshot nodes.

We consider a content whose popularity is equal to 0.1 among Tier-2 and Tier-3 ASs; we assume Tier-1 ASs do not hold a copy of the content. Furthermore, we run the emulator by restricting location requests to only Tier-2 and Tier-3 ASs. To summarize, Tier-1 ASs participate in the search process but do not contribute any further.

B. Sensitivity to lookup parameters

We characterize the performance of `link` by relying only on customers, i.e., parameters β and γ are both equal to 0. In this case, Tier-3 ASs can successfully resolve the location request only if they hold a copy of the content. Tier-2 ASs can exploit more search path, instead. Indeed, their P_{hit} is higher than the content popularity and the overall results are represented in Table I. It can be noted that increasing α raises P_{hit} from 0.105902 to 0.151068: the performance for $\alpha = 1$ represents an upper bound on the achievable performance. Furthermore, the rather small values of ALL for all considered values of α show that a little number of search requests contacts more than one CNS.

To evaluate the impact peers in the AS snapshot we consider all combinations of parameters α and γ where $\beta = 0$ in results we present in Table II. It can be noted that parameter γ has moderate impact since Tier-3 ASs have very limited peer AS relationships. On the contrary, Tier-2 ASs can exploit their

α	P_{hit}	ALL
0.1	0.105902	1.05899
0.5	0.124956	1.25496
1.0	0.151068	1.52305

TABLE I
VALUES OF P_{hit} AND ALL DURING THE LOOKUP AS FUNCTION OF α AND FOR $\beta = \gamma = 0$.

(α, γ)	P_{hit}	ALL
(0.1,0.1)	0.137882	1.44823
(0.1,0.5)	0.172990	1.94331
(0.1,1.0)	0.246075	2.68687
(0.5,0.1)	0.162374	1.66939
(0.5,0.5)	0.204280	2.20428
(0.5,1.0)	0.279786	2.64804
(1.0,0.1)	0.184314	1.83321
(1.0,0.5)	0.225457	2.22337
(1.0,1.0)	0.298151	2.85454

TABLE II
VALUES OF P_{hit} AND ALL DURING THE LOOKUP AS FUNCTION OF PAIR (α, γ) FOR $\beta = 0$.

(α, β)	P_{hit}	ALL
(0.1,0.1)	0.107914	1.08599
(0.5,0.1)	0.126893	1.27774
(1.0,0.1)	0.153042	1.54341
(0.1,0.5)	0.562925	6.73632
(0.5,0.5)	0.594792	6.86407
(1.0,0.5)	0.613455	6.77303
(0.1,1.0)	0.992233	9.1566
(0.5,1.0)	0.993146	9.80686
(1.0,1.0)	0.993425	10.9659

TABLE III
VALUES OF P_{hit} AND ALL DURING THE LOOKUP AS FUNCTION OF PAIR (α, β) FOR $\gamma = 0$.

popularity	P_{hit}	ALL
0.1	0.58548	6.90985
0.01	0.480732	41.2545
0.001	0.060271	134.144

TABLE IV
VALUES OF P_{hit} AND ALL DURING THE LOOKUP AS A FUNCTION OF POPULARITY FOR PARAMETERS $(\alpha, \beta, \gamma) = (0.1, 0.75, 0.1)$.

peering relations to increase their P_{hit} although the maximum achievable performance is just 0.298151.

The impact of parameter β on the performance of link is remarkable, instead. It can be noted from results in Table III that by increasing β from 0.1 to 0.5 for a very low value of α (0.1) we obtain $P_{hit} = 0.562925$ (from the value 0.107914). By further increasing it to 1 we obtain that location requests are successfully served almost surely for any value of α. Of course, this improvement is paid by the increased cost of the service in terms of the ALL values.

The last set of results we present is to analyze how the content popularity impacts on the cost of lookups and how effectively link is able to successfully serve location requests. To this end, we considered the triple of parameters $(\alpha, \beta, \gamma) = (0.1, 0.75, 0.1)$ and performed location requests for increasingly rare objects. We chose these low values for link parameters because it aims at avoiding that the search phase (and as a byproduct the content exchange) indiscriminately jumps on the different network locations thus possibly increasing transit fees.

Table IV shows results that link yields values of P_{hit} that are order of magnitudes higher than content popularity even for rather scarce object diffusion. Of course, the scarcer the content the higher the number of CNS to contact before finding one that owns a copy. Indeed, ALL values increase as content popularity decrease ASs although the average lookup length for the scarcer content is only 0.2% of the size of the AS snapshot we use for experiments.

As a final remark, please note that although the analysis we presented does not account for the dynamic evolution of the content popularity (i.e., we are assuming here that the content popularity does not change during the lookup phase), the insight it provides can be used by the CNSs to explore a wide set of parameters vs. the content popularity. In particular, performance can be tuned by letting each CNS modulate the costs of the lookup phase in terms of number of explored CNSs (and hence of the distance in terms of AS hops). In other words, the lookup algorithm can modulate the CNS's network awareness by tuning parameters (α, β, γ) to balance costs and expected P_{hit} since each CNS is aware of its connectivity relations and of the transit costs related with these relations.

C. Sensitivity to deployment of CNS

Here we evaluate the performance of link as a function of how widespread CNS are in the entire network. To this end, Table V shows results when Tier-1 ASs deploy a CNS with a certain probability P_{dep} for $(\alpha, \beta, \gamma) = (0.1, 0.75, 0.1)$ and content popularity equal to 0.1. It can be noted the contribution of Tier-1 ASs to the performance of link is not so high. Indeed, when Tier-1 ASs do not cooperate during the lookup we obtain $P_{hit} = 0.461884$ that is moderately less than the highest possible value, i.e., 0.58548. This can be explained by noting that Tier-2 ASs are generally well connected with many peers and many customers. This means that link can easily give up Tier-1 ASs and still be able to provide very good chances to successfully locate objects.

We further consider the case where no Tier-1 AS deploys a CNS and both Tier-2 and Tier-3 cooperate with probability P_{dep}. Results are summarized in Table VI; it can be noted that at least 40% of ASs should deploy a CNS to obtain a hit probability value that is greater than the content popularity.

The last set of results characterizes a system where Tier-1 ASs do not cooperate while all Tier-2 ASs do. We consider varying levels of cooperation of Tier-3 ASs (the majority of ASs in the CAIDA snapshot) modeled by the adoption probability P_{dep}. Results are reported in Table VII; they show that link performance are only slightly degraded when only 10% of Tier-3 ASs cooperate in the search process by adopting a CNS. This means that adoption of a CNS can progressively start by incentivizing Tier-2 ASs to participate in the lookup framework.

P_{dep}	P_{hit}	ALL
1.0	0.58548	6.90985
0.5	0.538563	6.3644
0.1	0.4622	4.64815
0.0	0.461884	4.63133

TABLE V
VALUES OF P_{hit} AND ALL DURING THE LOOKUP AS A FUNCTION OF P_{dep} FOR PARAMETERS $(\alpha, \beta, \gamma) = (0.1, 0.75, 0.1)$ AND POPULARITY 0.1.

P_{dep}	P_{hit}	ALL
1.0	0.461884	4.63133
0.5	0.179639	2.08642
0.4	0.127507	1.7857
0.1	0.0293718	1.11851

TABLE VI
VALUES OF P_{hit} AND ALL DURING THE LOOKUP FOR UNCOOPERATING TIER-1 ASS, AS A FUNCTION OF P_{dep} FOR PARAMETERS $(\alpha, \beta, \gamma) = (0.1, 0.75, 0.1)$ AND POPULARITY 0.1.

P_{dep}	P_{hit}	ALL
1.0	0.461884	4.77133
0.5	0.42685	4.63186
0.1	0.393548	4.58351

TABLE VII
VALUES OF P_{hit} AND ALL DURING THE LOOKUP FOR COOPERATING TIER-2 ASS, AS A FUNCTION OF COOPERATION OF TIER-3 ASS (PROBABILITY P_{dep}) FOR PARAMETERS $(\alpha, \beta, \gamma) = (0.1, 0.75, 0.1)$ AND POPULARITY 0.1.

IV. FURTHER DEVELOPMENTS

This section presents some improvements and features that could be explored and included in CNS service.

Discovery improvements.

1) To improve queries hit and limit messages, CNS can put in a cache the result of a successful query lookup giving positive results not in the current AS. Caching local pointers instead of the real data has many advantages: in particular, decreases the size of the caches in CNS servers, promotes local awareness, mobility and nomadism, and reduces CNS server overhead. Moreover, CNS discovery process promotes local data republishing contents whose owner or purveyors are far away in the CNS belonging to the current AS. The positive effect of caches applied to all the CNS databases can leverage the number of message exchanges between CNSes. Caches can be also useful in case of hypernames catching the interest of a large number of clients, producing unexpected overloading of CNS-traffic (also known as "flash crowd");
2) To reduce traffic and flooding attacks, each CNS can limit the number of `link` packets arriving from a customer or a provider or a peer; their number can be fixed on a AS-to-AS basis;
3) To improve liveness, a liveness politics can be implemented (as in Kademlia's bucket-table ordering republication [MM02]). Each publication in an authoritative CNS can have a lifespan: after the end of the lifespan, either the publisher re-publish the content in the CNS, or the record is simply dropped out from the CNS;
4) To limit the research space, a TTL can be introduced in `link` messages; TTL allows to limit the lifetime of lookup messages. A TTL counter attached to each link message allows to flush messages whose counter has elapsed;
5) To improve participation, incentives to locally republish contents retrieved abroad can be introduced: republication can be a simple pointer to another CNS. A tit-for-tat strategy could be installed between clients (looking for contents) and purveyors (distributing the contents) were the CNS should play a special role being in the middle of the above two actors;
6) To improve load distribution, CNS can perform load distribution among replicated copies of a single content. If CNS tables map a hypername into a lists of IP, then the CNS can respond with the entire list of purveyors, or it can rotate the ordering of the addresses within each reply. As such, IP rotation performed by CNS can distribute among multiple purveyors;
7) To improve the discovery success rate and focus the discovery search, each CNS can dynamically refine their α, β, and γ flooding parameters by combining with the success probability of a given tag in the previous queries.

Content aggregation in CNS. The data quality can be compromised by many factors, including data entry errors (*OpneOffice* instead of *OpenOffice*), missing integrity constraints (*"eat before December 12018"*), multiple convention "1^{st}, rue Prés. Wilson Antibes", vs. "1, rue du Président Wilson, Antibes"), optional arguments (+33(0)678358088 vs. 0678358088), see [EIV07] for a survey of data deduplication techniques. As a simple intuition, let the following hypername:

```
HN1 = name1:tag1:host1:fingcont
```

be published in some CNS and let

```
HN1 = name2:tag2:host2:fingcont
```

be retrieved by a `link` query: `HN1` and `HN2` differ in content names and in all logical attributes but the digital signature of the content `fingcont`, which is the same. Because the digital signature is the same, the two hypernames should be merged into a single one. More generally, each time a purveyor publishes an immutable content with a given `HN2`, or a query return a list of purveyors, the authoritative CNS should verify that the same content is not already published with a similar but syntactically different `HN1`[8] and, when it is the case, merge the two entries. Content aggregation should rewrite the previous two entries and substitute with the following ones:

```
HN1 = link to HN3
HN2 = link to HN3
HN3 = name1|name2:tag1,tag2:host1,host2:fingcont
```

[8] E.g. synchronizing mail or telephone contact across multiple google accounts.

where the symbol "`,`" denotes list concatenation and the symbol "`|`" denotes an "or" operator that allow to match both content names in pattern matching.

Mobility. Since traffic from wireless and mobile devices has exceeded traffic from wired devices [CIS17], most contents are requested and delivered by both wireless and mobile devices. It is well known that wireless and mobile devices may easily switch networks, changing their IP address and thus introducing new communication modalities based on intermittent and, possibly, opportunistic connectivity [XVS+14]. The CNS service discovery should be able to deal with mobility in case the owner/purveyor is a mobile host.

Nomadism. When a mobile node wants to publish a content, two cases can happen according to the (im)mutability of the content:

- *immutable:* (most common of the two). The authoritative CNS related to the mobile ISP accept the publication of an immutable content by a mobile user with the proviso of (i) recording the identity of the user, via e.g. the MAC address of the mobile device (or another identifier of the mobile node), and (ii) asking to the mobile user to republish the content more frequently than a fixed device, and (iii) possibly blacklisting a mobile device that appear and disappear too fast or too often.
- *mutable:* it deal with the possibility to keep an identity also in case the user is navigating through different mobile networks. The authoritative CNS related to the mobile ISP could accept the publication of a mutable content if and only if the logical attribute `fing_princ` is present and the logical attribute `host_list` contains only one symbolic name or only one IP.

Security. Until now, a few significant DNS attacks has corrupted the DNS service: this is because (i) DNS servers are machines managed and protected by system administrators, (ii) the DNS protocol pushes lookup always "below" the hierarchical database, minimizing the "uphill ascents", and (iii) of making use of cache techniques. We think that the above arguments could be applied also the CNS service because (a) the relatively fixed number of CNS servers ($\sim$70K) could be managed by AS system administrators, (b) `link` always pushes the location process first downhill the customer, and, only in case of failure, uphill through a peer or a provider. Nevertheless, the CNS discovery service is not vaccinated by either DDoS bandwidth-flooding attack, or man in the middle attack, or poisoning attack, or spoofing an IP of a node below an authoritative CNS.

References

[BCA+12] M. F. Bari, S. Chowdhury, R. Ahmed, R. Boutaba, and B. Mathieu. A survey of naming and routing in information-centric networks. *Communications Magazine, IEEE*, 50(12):44–53, 2012.

[CAI16] CAIDA. Center for Applied Internet Data Analysis: AS relationship. http://www.caida.org/data/as-relationships/, 2016.

[CIS17] CISCO. Cisco Visual Networking Index: Forecast and Methodology, 20162021. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html, September 2017.

[EIV07] Ahmed K. Elmagarmid, Panagiotis G. Ipeirotis, and Vassilios S. Verykios. Duplicate record detection: A survey. *IEEE Trans. on Knowl. and Data Eng.*, 19(1):1–16, 2007.

[Gao01] Lixin Gao. On inferring autonomous system relationships in the internet. *IEEE/ACM Trans. Netw.*, 9(6):733–745, 2001.

[GKR+11] Ali Ghodsi, Teemu Koponen, Jarno Rajahalme, Pasi Sarolahti, and Scott Shenker. Naming in content-oriented architectures. In *Proceedings of the ACM SIGCOMM Workshop on Information-centric Networking*, ICN '11, pages 1–6. ACM, 2011.

[JST+09] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs, and Rebecca L. Braynard. Networking named content. In *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*, CoNEXT '09, pages 1–12, New York, NY, USA, 2009. ACM.

[KCC+07] T. Koponen, M. Chawla, B. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica. A data-oriented (and beyond) network architecture. *SIGCOMM Comput. Commun. Rev.*, 37(4):181–192, 2007.

[MM02] Petar Maymounkov and David Mazières. Kademlia: A peer-to-peer information system based on the XOR metric. In *Peer-to-Peer Systems, First International Workshop, IPTPS 2002, Cambridge, MA, USA, March 7-8, 2002, Revised Papers*, pages 53–65, 2002.

[Moc83] P. Mockapetris. Domain names - concepts and facilities. https://tools.ietf.org/html/rfc882, 1983. RCF 883, updated 973, 1034, 1035.

[RL95] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP-4). https://tools.ietf.org/html/rfc4271, 1995. RCF 4271, obsoletes 1654, 1267, 1163, 1105.

[SBL+16] Wentao Shang, Adeola Bannis, Teng Liang, Zhehao Wang, Yingdi Yu, Alexander Afanasyev, Jeff Thompson, Jeff Burke, Beichuan Zhang, and Lixia Zhang. Named data networking of things (invited paper). In *First IEEE International Conference on Internet-of-Things Design and Implementation, IoTDI 2016, Berlin, Germany, April 4-8, 2016*, pages 117–128, 2016.

[XVS+14] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, Xe. Vasilakos, K. V. Katsaros, and G. C. Polyzos. A Survey of Information-Centric Networking Research. *IEEE Communications Surveys & Tutorials*, 16(2):1024–1049, 2014.

[ZAB+14] Lixia Zhang, Alexander Afanasyev, Jeff Burke, Van Jacobson, kc claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. Named data networking. *Computer Communication Review*, 44(3):66–73, 2014.

[Zoo03] Zooko Wilcox-O'Hearn. Names: Decentralized, secure, human-meaningful: Choose two. http://zooko.com/distnames.html, 2003.