LogNet: Extending Internet with a Network Aware Discovery Service

Luigi Liquori, Matteo Sereno

LogNet: Extending Internet with a Network Aware Discovery Service

Everything needs to change, so everything can stay the same *

[Extended abstract]

Luigi Liquori
Inria Sophia Antipolis Méditerranée, France
Luigi.Liquori@inria.fr

Matteo Sereno
Università di Torino, Italy
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 LogNet, a conservative extension of the current TCP/IP hourglass Internet architecture, that provides a new network aware Content Discovery Service.

Contents are referred via the new notion of *HyperNames* (HN), whose rich syntax allow to specify, hosts, pki, fingerprint and a large list of *optional logical attributes* (tags) attached to the content name, such as e.g. mutable vs. immutable contents, digital signatures, owner, availability, price, etc. Hypernames are in part human-readable and in part machine-readable and thus self-certifying.

Publication and discovery of HN is achieved using the new service *Content Name System* (CNS) with related protocol, whose behavior and architecture is, partly, inspired to DNS, and whose "routing logic" uses the BGP inter domain routing information. Moreover, the CNS service is implemented via a distributed protocol organized throughout a set of communicating CNS servers.

The core of CNS is the *Hypername Lookup Algorithm* (HLA) which "tunes" content discovery of being network aware, by taking advantage of the Autonomous System (AS) relationships. In particular, the HLA starts the content discovery process in the local AS (i.e., where the query starts), and in case of negative answer, propagate the query by accounting for the AS-to-AS relationships (i.e., peering-to-peering, provider-to-customer, customer-to-provider). After discovered, the contents are distributed with the classical distribution protocols.

Keywords

Content Discovery and Distribution, Network Awareness, Information Centric Networks, Naming.

*From "The Leopard" by Giuseppe Tomasi di Lampedusa.

ACM ISBN –.
DOI: -

1. INTRODUCTION

Information Centric Networks (ICN) is a clean-state approach to redesign the actual Internet infrastructure from an "host-centric", fully connected, paradigm to a "name-centric", loosely connected, paradigm where the focus is on named data instead on 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 main schools of thought: (i) *Content Centric Networks* referring to the Jacobson-based vision [18], where routing is driven by fully qualified - human readable - hierarchical names; (ii) *Data Oriented Network Architecture* (DONA) referring to a flat, unreadable but unique name-space [20] (see also [1, 2, 21, 29, 32] and the references therein).

Clean-state Design vs. Evolution of the Existing Network Architecture. While it is always exciting design 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 (or, even worst, new) network's equipments. Some lines of thought claim that it would be "reasonable" to envisage an IP dismission in the future definition of ICN.

This paper strongly supports the evolutive research line. Let us consider a programming language analogy: machine code was used at the very beginning of computers science by humans as the only low level programming language but later still employed by compilers as a target language to which map high level programming languages. We believe that the current TCP/IP-based transport protocol could be considered as a target protocol suite to efficiently map more sophisticated protocols and services. It is authors' believe that a lot of the above cited ICN proposals, including the one presented in this paper, could (and should) be deployed by a small, surgical, conservative extension on the current TCP/IP hourglass Internet infrastructure.

The network aware discovery service. Since the very beginning of this research vein, the *querelle* has been about

about how build a new kind of Internet based on *"Contents-as-Names"* instead of the current *"Names-as-IPs"* paradigm. The simple recipe developed in this paper is to forge a new paradigm, namely *"Contents-as-IPs"*.

This paper presents a lightweight Internet service to be implemented on the existing Internet stack, and more precisely, between the Transport and the Session Level (referring to the ISO-OSI protocol layering). We call this new service *Content Name System* (CNS) organized throughout a set of communicating CNS servers. In a nutshell, the purpose of this 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. In other words: the service binds, in the distributed CNS, set of IP-addresses to content-names, the latter modeled by HyperNames (HN). 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.

As the reader can suppose, each CNS will naturally be equipped with a *cache* (or more precisely with a database) containing for each queried content-name, the set of corresponding IP ordered by local awareness: caching local pointers instead of the real data has many advantages. In particular, it decreases the size of the caches in CNS servers, promotes local awareness, mobility and nomadism, reduces CNS server overhead, etc. Note that once the CNS reply to a query giving a list of IP that are owners or purveyors of some contents, the proposed protocol terminates and the real transfer of the data proceeds using usual client-server or peer-to-peer protocols with the positive consequence that the choice of the IP "actors" has been done promoting locality awareness and respecting inter-domain routing.

Moreover, the CNS service promotes data republish, by local republishing contents whose owner/purveyors are far away in the CNS belonging to the current AS. Therefore, we could imagine as a potential use of that service, well-known peer-to-peer protocols, like BitTorrent [9], performing location-aware file exchange between nodes belonging to same/close ASes or, better, using some unofficial BitTorrent extensions, like e.g. the BitTorrent Location-aware Protocol 1.0 [26], or even better raising the CNS servers also to the status of "location aware BitTorrent trackers".

One last remark: nowadays many P2P protocols saturate the network traffic by an "hysteric" (from the point of view of network providers) flow of network connections from one AS to another AS without taking care of the fact the Internet traffic have a *real monetary cost* caused mostly by following BGP routes and AS-IS relationship between ASes. This fact, probably and also the other well-know fact that mostly of the exchanged contents breaks the DMCA [25] caused to the demonization of almost all P2P protocols. In this paper we try to capitalize (i) the P2P experience, and (ii) the experience in studying the AS-relationships (i.e., provider-to-customer, customer-to-provider, peering relationships) [3, 15], to propose services and protocols that can be easily included in the current Internet infrastructure.

Content Name System (CNS). Is a hierarchical and decentralized naming system providing a new Internet service that translates content names into IP addresses needed to later retrieve the content itself. The CNS servers are distributed over the ASes, and more precisely there is at least one CNS for each AS taking care of resources registered inside the AS itself. In a nutshell, the CNS server hierarchy mimics the AS relationship hierarchy. More precisely, the CNS server associated to a given AS set its position in the CNS server hierarchy starting from the (private) AS business relationships. As we know, a nice approximation of those relations is snapshotted every month by CAIDA [6].

HyperName (HN). It denotes a name of one content. The name is composed by an 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 the content, the ownership, its integrity, its signature, its price, availability, etc. Attributes are not enforced to be human readable. HNs are used essentially to *publish* contents on a local CNS and to *resolve* the HN with the list of IP addresses of the owners or the purveyors of the data. HN does not enforce to denote uniquely neither the content neither the owner or the purveyor of the content, but this can be enforced by using logical attributes.

Frequently Asked Questions. In the last decade, Information Centric Networking research has produced, a lot of very interesting proposals. These proposals (or some of them) will be the basis for future network architectures. On the other hand, in this paper we will use some of the ideas developed by the ICN research venue to propose a content discovery service that can be implemented in the current Internet and that (hopefully) can alleviate some of the problem in the content distribution process. This subsection try, in a somewhat informal, to give some insight to the LogNet proposed architecture by means of **Q**-**A**.

- **Q**: Does LogNet can be considered as a conservative extension of the actual Internet?
  A: Yes: LogNet do not enforce to use the new CNS service: as example, an happy user of BigG services and P2P protocols can continue to stay with it without changing their habit.

- **Q**: Does LogNet architecture is based on a clean-slate design?
  A: LogNet is not based on a clean-slate design: it is small, surgical extension of the current OSI stack providing a new service allowing to publish and to discover locally aware contents by HyperNames. LogNet is a simply a modest and conservative extension of the current Internet.

- **Q**: Does LogNet can be seen as a sort of Distributed, P2P, BigG killer ?
  A: We should say no ... in order to get invited for a tech-talk at Mountain View ;-) More seriously, we don't think so. Since LogNet enforce at least one CNS being hosted in each AS, it follows that the ability of each node to publish a content and then retrieve another is related to the HLA, taking an HN as a query input and producing, in case of a successful query, either the full content, or a non-empty set of IPs – where the full content would be retrieved – or a failure. Nevertheless, we could say that LogNet architecture share the same BigG vision/mission (i.e. *to provide access*

to the world's information in one click/to organize the world's information and make it universally accessible and useful) but in a fully distributed fashion.

- **Q**: Does LogNet architecture is a way to decentralize web engines?
  A: We don't think so: the advanced technology achieved by the most famous web search engine will guarantee to "stay in peace" for the next two decades, at least. It is true that as soon as the (open source) CNS codebase will be released, and because of the fundamental distributed nature of the HLA algorithm many simple and effective optimizations could appear in a relatively short term focusing on contents published via the HTTP protocol.

- **Q**: Does LogNet is *yet another* overlay network proposal?
  A: Of course this is not the case. LogNet is not an application layer overlay network: it is a *network aware content discovery service* that do not introduce another logical address space disconnected with the IP one. To do this, the LogNet architecture extend the Session Layer.

- **Q**: Does LogNet have some analogies with CDN, à la Akamai?
  A: No. CDN is an overlay network whose mission is *to help server providers to move and cache popular contents in order to leverage busy servers*: mirroring is done using an overlay network on the top of the actual Internet. LogNet is an extension of the current Internet architecture: its mission is orthogonal to CDN, namely *to help clients to locate the closest copy of the required content in the current (or closest) AS*.

- **Q**: Does LogNet add "network awareness" to classic peer-to-peer protocols and video streaming ?
  A: Possibly yes: the exact amount of network awareness needed to be carefully exploited in order to enhance scalability of P2P protocols in the current Internet.

- **Q**: By globally deploying the CNS service, should we expect a general overcrowding of the BGP routes?
  A: No because : (i) the traffic generated only by content discovery is not exhaustive by construction; (ii) the use of caching techniques (as the one used for DNS) that limit global searches, and, (iii) byproduct, the resulting content distribution will limit future CNS queries.

- **Q**: By using the IP resolved by a CNS query to move contents from IP to IP, should we expect a general overcrowding of the BGP routes in the successive content distribution?
  A: No. Very informally, the CNS query will explore the CNS hierarchy by performing the search flow always from local first, then through customer-AS, then through peering-AS, and finally through provider-AS. If the content is found, the content distribution will move from the owner/purveyor to the demander preserving, at the best effort, the well known *no-valley routing* intra AS [14, 15, 27].

- **Q**: By globally deploying the CNS service, should we expect some decrement of the *global number of IP packets*?
  A: Uhm, quite difficult question: surely the LogNet extension force contents to be discovered and cached in the AS where they are looked for, and offer always to P2P protocols the "closest" local peers. Maybe a new kind of *"Network Aware P2P"*?

- **Q**: The CNS protocol suite should be available at which network level?
  A: In principle, all services above the CNS protocol could take advantage of the discovery service, and especially services at the Application Level, such as e.g. HTTP, FTP, and SMTP, and all the P2P ones.

- **Q**: The CNS protocol suite could call which protocol?
  A: In principle, all services below it can be called: among others, DNS at the Session Level, and BGP and TCP-UDP at the Transport Layer.

- **Q**: The CNS service share some similarities with the DNS service?
  A: Just a bit, the HLA share some similarities with bind, but both works on different topologies, remember that the DNS hierarchy is not network-aware while the CNS hierarchy must follows the AS "upstream-peering-downstream" topology.

Related Work. Without any doubt, we are respectfully and genuinely indebted with all the literature cited in this paper: first of all the generous tutorials and internet draft, which allowed to warm us; then all the nice ICN proposals we have read ([5, 12, 17, 19, 22, 28, 30] in addition to the previous cited ones). Our objective was not to set up yet another proposal but try to get the best of every architecture and proposed protocol, and having in mind that we wanted to build a conservative extension of the actual IP hourglass architecture. Finally, we were also inspired to a logical network architecture, called ARIGATONI, we developed some year ago, featuring resource discovery and virtual intermittence protocols [4, 7, 8, 10, 11, 23]. CAIDA maps were also more than useful, and DNS protocol was also source of great inspiration.

Plan of the paper. Section 2 introduces HyperNames syntax and the CNS service and topology: the section also presents a first version of the Hypername Lookup Algorithm (HLA); Section 3 present some preliminary simulations to asses the performance of CNS service; Section 4 presents some further developments.

2. HYPERNAMES AND CONTENT NAME SYSTEM

HyperNames. A LogNet's *HyperName* is a name string enriched with a number of (optional) parameters helping to identify a content, and possibly its ownership, its integrity, its hosting, and its attribute-list. More formally:

DEFINITION 2.1 (HYPERNAME). *An* HyperName *(HN) is generated by the following abstract syntax:*

`[fing_princ:][fing_cont:][hosts:][tags:]cont_name`

where `cont_name` is a mandatory, human readable, string denoting a content name, `tags` is the optional list of *tags* associated with a given content, `hosts` is the optional list of *hostnames* being the purveyors of the content, `fing_cont` is the optional digital signature (i.e. the cryptographic hash) of the content, `fing_princ` is the optional digital signature (i.e. the cryptographic hash) of the public asymmetric key of the *principal*, i.e. the owner of the content.

In short, an HN is an human-readable content name followed by some optional human-readable tags and then by some optional machine-readable datas. A HN *human view* can display just the content name and the tag list, leaving all the others machine-readable datas to the *internal view*. By using optional prefixes in HyperNames, we greatly contribute to identify, search and retrieve contents and ownership in the LogNet architecture, to normalize and optimize the CNS distributed data base using *Data Deduplication techniques* [13] (see in the last section) that can be used to improve the performance of the the HyperName Lookup Algorithm (HLA) presented later in this section. The concept of HN as a name with "different views" is inspired by the Zooko triangle and by all the literature and proposals dealing with content names [16, 33].

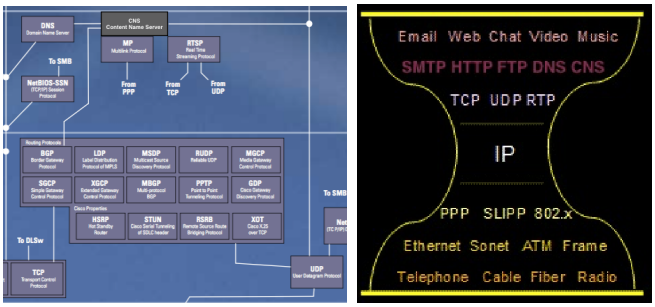

Figure 1: Zoom of network protocols "close" to CNS and new hourglass IP model

Content Name System. In terms of protocol stack the CNS service can be located in the IP hourglass at the same level of the DNS, as Figure 1 pictorially shows. In particular, the CNS provides a *new* core Internet service, namely translating *HyperNames* (HN) to lists of IP addresses: in math style: $\text{HN} \Longrightarrow \{\text{IP}_\text{i}\}^{\text{i}\in\text{I}}$, with the set I possibly empty in case of content discovery failure. As we did in the Introduction, we choose to present the main features of the CNS service by putting it face-to-face with DNS service.

- The DNS [24] is a fundamental "phone book"-like 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 internet hosts. The CNS performs a distributed, network aware, content discovery service for the Internet. Similarly to the DNS, the CNS uses the UDP protocol to query other distributed CNS servers to answer client session like *"find an IP addresses that (i) published a content matching the hypername HN and (ii) whose AS is "network-close" to the AS of the requester"*. Therefore, CNS service also provides information about the hosts that have published a given content [31];
- The DNS delegates name resolution into *Domain Zones* from the smallest to the biggest zone. With the same idea, the CNS delegates content discovery (content name resolution) through *Autonomous Systems* (AS) always trying to follow, whether possible, a "reverse cash flow" route in order to suggest to the further content delivery an ordinary "cash flow" route;
- The DNS distributed database is indexed via domain names. On the other hand, the relations among CNS servers are derived by the relations among the Autonomous Systems (i.e., customer-to-provider, provider-to-customer, peering relations). These relations can be derived by using CAIDA's "AS Relationships Dataset" maps (see [6] and Gao's pioneering work of [15]);
- The DNS queries can be iterative or recursive: the same holds for the CNS, while, as for the DNS, iterative queries are preferred for efficiency reasons.

CNS' Hierarchical Topology. To scale up, the content-based distributed database is organized into a *hierarchy of servers* distributed according to the classical Tier-1/Tier-2/Tier-3 AS topology (as described by the AS-to-AS relationship datasets of CAIDA [6]). As well explained by CAIDA, an annotated AS/ISP graph includes relations of kind customer-to-provider (or, symmetrically, provider-to-customer) and of kind peer-to-peer. Customer-to-provider AS relation refers to a relation where the customer ISP pays the provider ISP for transit (the, so called, *flow of money*). Therefore, ASes at lower levels pay ISPs at higher levels in exchange for access to the rest of the Internet. A peering link, instead, connects two ASes who have agreed to exchange traffic on a *quid pro quo* basis. ASes involved in a peering relation exchange traffic only between each other and each other's customers.

Each AS must have at least one CNS, called *authoritative*, whose database will take into account the association of HN with a list of IPs that have registered a content named by an HN. The authoritative CNS also knows exactly its position in the distributed database, namely (i) the IP addresses of all customers CNS, (ii) the IP addresses of all providers CNS and (iii) the IP addresses of all peer-to-peer CNS: this will allow to dispatch queries along the distributed database.

Content Publication. In a nutshell, in order to make a content "discoverable", a content must be first "published" by some owner or purveyor: this is done simple by sending to the authoritative CNS a message of the shape

`send publish(HN) to CNS`

where `HN` is the HyperName associated with the given content and the sender is the owner or the purveyor of the content. Note that the publication in a CNS associate an HN with a principal, and that principal holds the content as an owner or a purveyor (the content being mutable or immutable).

More precisely, suppose a given content C be available by an host belonging to an Autonomous System AS: the

```
1.01 on receipt of links(HN,DOWN) from provider do                          receive a query from a ''downhill''
1.02 value = lookupdb(HN);                                                  search HN in the CNS' local data base
1.03 if (value ≠ 0)                                                         some IP publishing HN are found
1.04 then return value to provider;                                         return those IP ''back to the downhill''
1.05 else list = select(α,customerlist);                                    select some customers CNS
1.06      forall cus ∈ list do value = value ∪ send links(HN,DOWN) to cus;  and forward the query downhill through a customer
1.07      return value to provider;                                         return the CNS list (can be empty) ''back to the hill''
```

Figure 2: LINKS: query from downhill continue on α thread downhill

```
2.01 on receipt of links(HN,UP) from peer do                                receive a query from a peer on the ''top of the hill''
2.02 value = lookupdb(HN);                                                  search HN in the CNS' local data base
2.03 if (value ≠ 0)                                                         some IP publishing HN are found
2.04 then return value to peer;                                             return those IP ''back to the top of the hill''
2.05 else list = select(α,customerlist);                                    select some customers CNS
2.06      forall cus ∈ list do value = value ∪ send links(HN,DOWN) to cus;  and forward the query but downhill through a customer
2.07      return value to peer;                                             return the CNS list (can be empty) ''back to the top of the hill''
```

Figure 3: LINKS: query from uphill being on the top of the hill will change on α thread downhill

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 HN that will be sent as a formal parameter to the authoritative CNS. Note that the host decides which attribute to attach the HN and if publish that content as a *owner* or as a *purveyor*. In the first case the publication is done by a simple write in the CNS' database (depending on a local policy, the CNS could ask to republish the content every n seconds). In the second case the host could be asked to "package" a `.torrent`[1] file and write it in the CNS' database; in the BitTorrent jargon the purveyor play a role of "seed" and it will be asked to republish itself as a purveyor of the content C every m seconds. Further nodes entering the swarm for C will be asked to publish his name in the torrent. In other word, for that content, the CNS server is playing a kind of *network aware BitTorrent tracker*.

Hypername Lookup Algorithm (HLA) in a nutshell. As said before, each AS holds an authoritative CNS server, that records the mappings for all the HNs published in the AS. To scale up, the CNS service is organized in a distributed hierarchical database, implemented in a hierarchy of servers. Each CNS interacts with the others through a distributed algorithm called *Hypername Lookup Algorithm* (HLA). In a nutshell, when a client asks for a given HN, the following actions are taken (note that we let, α, β, and γ being AS-specific parameters):

Step 1. the client first contacts its authoritative CNS and then searches the HN in the local publications (i.e. in the current and in the peering CNS);

Step 2. if the above fail, then the authoritative CNS forwards the query through α CNS belonging to ASes in downstream, i.e., with which we have signed some provider-to-customer agreement;

Step 3. if the above fails, then the authoritative CNS forward the query through γ CNS belonging to ASes in peer, i.e., with which we have signed some peer-to-peer agreement;

Step 4. if the above fail, then the authoritative CNS forward the query through β CNS belonging to ASes in upstream, i.e., with which we have signed some customer-to-provider agreement.

The LINKS pseudocode. A first HLA implementation is showed by the LINKS[2] pseudocode in Figures 2, 3, and 4.

Start of the "HLA". A client sends a query to the authoritative CNS server where the client belongs to, with argument the HyperName HN. Following the DNS jargon, this query is *recursive* in the sense that the client will be blocked until the CNS will answer positively with a result containing a set of addresses $\{\mathrm{IP_i}\}^{i\in I}$ associated with HN, or with a search failure.

Figure 2. This code refers to the general case when the current CNS receives a "links" message with an HN and a downhill direction from a provider-CNS (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[3] (1.04); else selects α customer-CNS (1.05) and sends α-*iterative* "links" queries with the same hypername and the same downhill direction (1.06); then collects the result value and send it back to the sender of the first links message (1.07).

Figure 3. Following the Gao jargon, this code refers to the well-known case of being "on the top of the hill", i.e., receiving a message from uphill and from a peer-to-peer-CNS. 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* "links" queries (2.06).

Figure 4. This code refers to the case where a CNS receive a "link " message from uphill from a customer-CNS. Again, following the Gao 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

[1]Note that the BitTorrent protocol is chosen just as an example of a distributed file sharing protocol.

[2]**L**ogNet **I**nternet **N**etwork **K**ey **S**earch.

[3]At the beginning of the search, the sender is just the authoritative-CNS itself, while in the middle of the HLA, the sender is a provider-CNS.

```
3.01 on receipt of links(HN,UP) from customer do                                   receive a query from a ''uphill''
3.02 value = lookupdb(HN);                                                        search HN in the CNS' local data base
3.03 if (value ≠ 0)                                                               some IP publishing HN are found
3.04 then return value to customer;                                               return those IP to ''back to the uphill''
3.05 else list = select(α,customerlist);                                          select some customers CNS
3.06      forall cus ∈ list do value = value ∪ send links(HN,DOWN) to cus;        and forward the query but downhill through a customer
3.07      if (value ≠ 0)                                                          some CNS are suggested
3.08      then return value to customer;                                          return those CNS ''back to the uphill''
3.09      else list = select(γ,peerlist);                                         select some peers CNS
3.10           forall per ∈ list do value = value ∪ send links(HN,UP) to per;     and forward the query uphill through a top of the hill peer
3.11           if (value ≠ 0)                                                     some CNS are suggested
3.12           then return value to customer;                                     return those CNS ''back to the uphill''
3.13           else list = select(β,providerlist);                                select some provider CNS
3.14                forall pro ∈ list do value = value ∪ send links(HN,UP) to pro;  and forward the query uphill through a provider
3.15                return value to customer                                      return the CNS list (can be empty) ''back to the uphill''
```

Figure 4: LINKS: A query from an uphill will continue on three directions: first α-downhill, then γ-downhill, and finally β-uphill

sender (3.04); else select α customer-CNS[4] (3.05) and send α-iterative "links " queries with the same hypername but *inverting the direction* from uphill to downhill (in other words: "repush" downhill the query) (3.06); in case of success, the result value is returned to the sender (3.08); else select γ peer-CNS (3.09) and send γ-iterative "links" queries with the same hypername and the same direction[5] (3.10); in case of success, the result value is returned to the sender (3.12); else select β provider-CNS (3.13) and send β-iterative "links" queries with the same hypername and the same uphill direction (in other words: go uphill only after 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.

After this short pseudocode-review, the reader has surely observed that:

Note 1. All messages not matching with the above three Figures are simply flushed by the receiving CNS.

Note 2. In case of α, β, and γ equals to 1, the number of CNS iterative-messages can be "modest", and that a careful tuning of those parameters is required to find a good trade-off between scaling/flooding and successful resource discovery. We'll see in the following, how we can improve the number of successful queries and limit the message flooding (1) by introducing ordinary techniques of "Caches" in CNS, and (2) by parsing optional parameters in HN, and (3) by introducing a "publication lifespan" in CNS publications, and (4) by introducing "TTL" in message queries, and (5) by introducing "incentives" to republish in the local AS contents retrieved abroad.

HLA Optimization 1: Adding Cache in CNS. In perfect analogy with DNS, a CNS could put in a cache the result of a successful query lookup giving positive results not in the current AS. The positive effect of a small caches (maintained in RAM) applied to all the CNS databases could leverage the number of message exchanges between CNS. Because the presence of a CNS mapping is no more permanent, CNS servers could also discharge cached records after a period of time to be fixed in the LINKS codebase. Caches are shown very useful in case of HN that catches the interest of a large number of clients, and get unexpected overloading of CNS-traffic (sometime called *"flash crowd"*).

HLA Optimization 2: Processing HN's Optional Parameters. As said before, an HN can have the following form:

```
[fing_princ:][fing_cont:][hosts:][tags:]cont_name
```

Until now, all optional parameters (logical attributes) were unused in the pattern matching algorithm hidden in the HLA algorithm. As such, a possible optimization should concern how to process the four logical attributes.

`tags:` A list of tags can be attached to an HN in order to choose α, β, γ CNS to which forward the query lookup. The purpose is to limit the search space and improve the success rate. As shown in the LINKS pseudocode, the choice of α, β, and γ is fixed once for all, but a better use of the list of tags can be done as follows: suppose that each CNS have a data structure called Tag_rate associating to each tag `t` a triple of probabilities $(prob^{t}_{\alpha}, prob^{t}_{\beta}, prob^{t}_{\gamma})$. The Tag_rate structure is updated each time a consumer-CNS, a peer-CNS, or a producer-CNS reply successfully for an HN tagged with `t`. The flooding parameters in the LINKS instructions (1.05,2.05,3.05,3.09,3.12) will be adjusted following the previous tag success rate. In few words, the lookup history can greatly customize and improve the CNS routing.

`hosts:` When an hostname list prefixes an HN, this means that the content name must be directly retrieved from one of the hostnames in the list. In this case, the local CNS just perform a DNS query to transform the given hostname into a single IP address and return, leaving the content transfer outside the scope of the CNS service.

`fing_cont:` This logical attribute allows to improve the HLA in case of immutable contents: when a content's fingerprint prefixes an HN, this means that the integrity of the content to be retrieved can immediately be verified as soon as we retrieve the content itself.

[4]Pay attention to not choose the customer-CNS that have sent the query.

[5]Pay attention that successives execution of code in Figure 3 will later invert the direction from uphill to downhill. In other words: "repush" downhill the query.

`fing_princ:` This logical attribute allows to improve the HLA in case of mutable contents of a unique owner: when a principal's fingerprint prefixes an HN, this means that when the client receive a content together with the public key of the owner, the identity of the latter can be immediately verified as soon as we retrieve the content itself.

HLA Optimization 3: Introducing a "Lifespan" in CNS publications. This simple optimization allows to better organizes CNS databases and to improve caches performance: it states that each publication in an authoritative CNS has 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.

HLA Optimization 4: Introducing a "TTL" in LINKS messages. This simple optimization allows to limits the lifetime of lookup messages. A "Time to Live" (TTL) counter attached to each LINKS message permits to "flush" messages whose counter has elapsed. This also limits message flooding in the HLA.

HLA Optimization 5: Adding "Incentives" to Locally Publish Messages. This simple optimization introduce some good habits to diffuse immutable contents in the distributed hierarchical database of CNS: it states that every client using the CNS discovery service, should get some incentives to "locally republish" some contents in case the discovery service answer returning a pointer to a content in another AS. 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 (business? content reputation?) role being in the middle of the above two actors.

Additional services provided by a CNS. The primary mission of a CNS is the one of translating HN into a set of IP addresses. Nevertheless, CNS could have other missions and in particular (i) *Content Aggregation* and (ii) *Load Distribution*.

Content Aggregation in CNS. This service concerns the aggregation of publication of two *semantically equal* contents (i.e. having the same `fing_cont`) with two *syntactically different* HN. More precisely, every time a purveyor publish an immutable content with a given HN2, the authoritative CNS can verify that the same content is not already published with a similar but equationally different HN1[6]. We explain the problem in short: if `fing_cont` is omitted in one HN, then there is no way to formally check equality between two contents having potentially different and "incompatible" tag lists. Note that the concept of "tag-incompatible" is a semantic one and not just syntactic. Indeed slightly different content names, differing, e.g., for tiny typographic errors and incompatible host name lists could refers to the *same real content*. This is the case where the CNS *Aggregation Algorithm* (A2) will try to merge some entries in order to keep the content table most faithful as possible (so dealing with *structural/lexical heterogeneity*)

Just to give an example of the many potential aggregations, let the following two different entries

```
HN1 = fing_cont:hosts1:tags1:cont_name1
HN2 = fing_cont:hosts2:tags2:cont_name2
```

be published in some authoritative CNS. The acute reader can see that HN1 and HN2 differs in content names and in all logical attributes but the digital signature of the content which is the same. A content aggregation will rewrite the previous two entries and substitute with the following ones:

```
HN1 = linksto HN3
HN2 = linksto HN3
HN3 =
fing_cont:hosts1,hosts2:tags1,tags2:cont_name1|cont_name2
```

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

Note that *Data deduplication* [13] is an emerging technology that introduces reduction of storage utilization: the study of A2 is out of the scope of this paper, and will be described in a future work.

Load distribution. As DNS does, CNS can perform load distribution among replicated copies of a single content. If CNS tables maps an HN into a *lists of IP sets*, then the CNS can respond with the entire list of nodes, or it can "rotates" the ordering of the addresses within each reply. As such, IP rotation performed by CNS can distributes the content distribution among multiple purveyors.

3. PRELIMINARY PERFORMANCE RESULTS

This section describes the first evaluation of the CNS service and of the LogNet Internet architecture extension. In particular, by using real ASes topologies provided by CAIDA [6], we evaluate the sensitivity of the lookup algorithm LINKS presented in the previous section with respect to parameters α, β, and γ.

For the experimental evaluation we select an ASes topology provided by CAIDA with 45427 ASes (i.e. a 2013 shapshot).

To this topology we apply a simple classification criterion for identifying Tier-1/Tier-2/Tier-3. We use this classification to distinguish the behavior of the lookup algorithm for the different type of ASes. An illustrative example of the differences among the different type of ASes is the role of the peering relations. It is well know that the peering relationships among Tier-3 ASes are (mainly) established for reducing transit costs, while at level of Tier-1 ASes the peering relationships are established for global connectivity purposes. Obviously, these differences must be accounted for the lookup algorithm. A typical case is when the CNS where the query is originated explores its peering neighborhood. In case of a Tier-1 CNS, it explores the whole set of the ASes with peering relationships with the CNS. On the other hand, in case of a Tier-2 CNS (or Tier-1), it performs a random selection on the peering CNSs according to the γ parameter.

In our experiment we evaluate the average exploration path length, that is, the average number of CNSs explored

[6]The "smarphone-addicted" readers have often experienced this in merging different contact lists.

during the search phase in case of search failure. In other word, we always account for the worst case of the query process. Table 1 summarizes the results obtained for different values of (α, β, γ). It can be seen by this preliminary set

(α, γ, β)	Average path length
$(1, 1, 1)$	17.68
$(1, 5, 1)$	34.10
$(1, 1, 5)$	114.85
$(5, 1, 1)$	71.64
$(5, 5, 1)$	98.75
$(5, 1, 5)$	247.81
$(10, 1, 1)$	207.50
$(10, 5, 1)$	257.43
$(10, 1, 5)$	410.73

Table 1: Average number of CNSs explored during the search phase as function of (α, β, γ)

of experiments that the average number of CNSs explored during the search phase is rather small. This is rather encouraging especially because we assume the worst search case (i.e., no content is published and no cache mechanism is implemented). We can see the different effects of (α, γ, β) parameters: in particular, we can compare the effects of these parameters and see that as soon as we increase the β (forward and flood to the provider-CNS), the number of hops will increase considerably.

4. FURTHER DEVELOPMENTS

This final section discuss mobility and security issues related with the LogNet Internet extension and the CNS network aware discovery service.

4.1 Mobility and Nomadism

Since traffic from wireless and mobile devices will probably exceed traffic from wired devices by 2016, we could expect that most contents could be 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" [32]. Since the LogNet architecture consists of adding a content discovery service to the current Internet, it follows that the difficulty resulting of dealing with mobility could arise especially in case the owner/purveyor is a *mobile host*. The symmetric case where the client is a mobile node can, without loss of generality, be considered out of the scope of the discovery service.

The Publisher is a Mobile node. In this case the mobile node wants to publish a content: two cases can happen according to the (im)mutability of the content:

Immutable: (this case being surely the most common of the two). The authoritative CNS related to the mobile ISP could 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 re-publish the content more frequently than a fixed device, and (iii) possibly "blacklisting" a mobile device that "publish and ~~perish~~ escape" too fast or too often.

Mutable: this case is less common but quite challenging since 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 `hosts` contains only one symbolic name or only one IP.

4.2 Security

DNS history teaches us that, up to date, there has been no significant DNS attacks that has successfully impeded the distributed DNS service: the secret of this success story was especially (i) because DNS servers are machines managed and "protected" by system administrators, and (ii) because the DNS protocol pushes lookup always "below" the hierarchical database, minimizing the "uphill ascents", and (iii) because of making use of well-known techniques of caching. Do CNS could have the same chance? We honestly don't know, a more deep study of this protocol is undergo: nevertheless the following indices make us to think positive: (a) the 70K+ CNS servers could be managed by AS system administrators and (b) the HLA always pushes routing first downhill the customer-CNS distributed database, and, only in case of failure, uphill through a peer-CNS or a provider-CNS. Anyway, in its current state, the CNS service is not vaccinated by well-known attacks like, DDoS bandwidth-flooding attack, or man in the middle attack, or poisoning attack, or spoofing an IP of a node below an authoritative CNS.

5. REFERENCES

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman. A survey of information-centric networking. *IEEE Communications Magazine*, 50(5):26–36, 2012.

[2] 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.

[3] G. D. Battista, M. Patrignani, and M. Pizzonia. Computing the types of the relationships between autonomous systems. In *Twenty-Second Annual Joint Conference of the IEEE Computer and Communications*, INFOCOM '03, pages 156–165. IEEE, 2003.

[4] D. Benza, M. Cosnard, L. Liquori, and M. Vesin. Arigatoni: A Simple Programmable Overlay Network. In IEEE, editor, *Modern Computing, 2006. JVA '06. IEEE John Vincent Atanasoff 2006 International Symposium on Modern Computing*, pages 82–91, 2006.

[5] M. Caesar, T. Condie, J. Kannan, K. Lakshminarayanan, I. Stoica, and S. Shenker. Rofl: Routing on flat labels. In *Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '06, pages 363–374. ACM, 2006.

[6] CAIDA. Center for Applied Internet Data Analysis: AS relationship.

http://www.caida.org/data/as-relationships/, 2016.

[7] R. Chand, M. Cosnard, and L. Liquori. Powerful Resource Discovery for Arigatoni Overlay Network. *Future Generation Computer Systems*, 24(1):31–48, 2008.

[8] R. Chand, L. Liquori, and M. Cosnard. Improving Resource Discovery in the Arigatoni Overlay Network. In *Proc of 20th International Conferences of Architecture of Computing Systems - ARCS* , volume 4415 of *Lecture Notes in Computer Science*, pages 98–111. Springer Verlag, 2007.

[9] B. Cohen. Bittorrent protocol specification v1.0. https://wiki.theory.org/BitTorrentSpecification.

[10] M. Cosnard and L. Liquori. Weaving Arigatoni with a graph topology. In C. S. Press, editor, *1st International Conference on Advanced Engineering Computing and Applications in Sciences ADVCOMP 2007*, pages 55 – 59, 2007.

[11] M. Cosnard, L. Liquori, and R. Chand. Virtual Organizations in Arigatoni. In *Proceedings of the Second International Workshop on Developments in Computational Models (DCM 2006)*, volume 171- issue 3 of *Electronic Notes in Theoretical Computer Science*, pages 55–75. Elsevier, 2006.

[12] A. Detti, N. Blefari-Melazzi, S. Salsano, and M. Pomposini. CONET: a content centric inter-networking architecture. In *2011 ACM SIGCOMM Workshop on Information-Centric Networking, ICN*, pages 50–55, 2011.

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

[14] N. Feamster, H. Balakrishnan, and J. Rexford. Some Foundational Problems in Interdomain Routing. In *3rd ACM SIGCOMM Workshop on Hot Topics in Networks (HotNets)*, 2004.

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

[16] A. Ghodsi, T. Koponen, J. Rajahalme, P. Sarolahti, and S. Shenker. Naming in content-oriented architectures. In *Proceedings of the ACM SIGCOMM Workshop on Information-centric Networking*, ICN '11, pages 1–6. ACM, 2011.

[17] M. Gritter and D. R. Cheriton. An architecture for content routing support in the internet. In *Proceedings of the 3rd Conference on USENIX Symposium on Internet Technologies and Systems - Volume 3*, USITS'01, pages 4–4. USENIX Association, 2001.

[18] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. 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.

[19] P. Jokela, A. Zahemszky, C. Esteve Rothenberg, S. Arianfar, and P. Nikander. Lipsin: Line speed publish/subscribe inter-networking. In *Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication*, SIGCOMM '09, pages 195–206. ACM, 2009.

[20] 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.

[21] D. Kutscher, S. Eum, K. Pentikousis, I. Psaras, D. Corujo, D. Saucez, T. C. Schmidt, and M. Waehlisch. ICN Research Challenges. Internet-Draft draft-irtf-icnrg-challenges-06, Internet Engineering Task Force, 2016. Work in Progress.

[22] D. Lagutin, K. Visala, and S. Tarkoma. Publish/subscribe for internet: PSIRP perspective. In *Towards the Future Internet - Emerging Trends from European Research*, pages 75–84, 2010.

[23] L. Liquori and M. Cosnard. Logical Networks: Towards Foundations for Programmable Overlay Networks and Overlay Computing Systems. In *Trustworthy Global Computing, Third Symposium, TGC 2007, Revised Selected Papers*, volume 4912 of *Lecture Notes in Computer Science*, pages 90–107. Springer, 2007.

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

[25] U. S. C. Office. The Digital Millenium Copyright Act of 1998 : U.S. Copyright Office Summary. http://www.copyright.gov/legislation/dmca.pdf, 1998.

[26] B. V. Petkantchin. Bittorrent location-aware protocol 1.0 specification. https://wiki.theory.org/BitTorrent_Location-aware_Protocol_1.0_Specification.

[27] S. Y. Qiu, P. D. McDaniel, and F. Monrose. Toward valley-free inter-domain routing. In *IEEE International Conference on Communications*, ICC '07, pages 2009 – 2016. IEEE, 2003.

[28] D. Smetters and V. Jacobson. Securing network content. Technical report, Palo Alto Research Center, 2009.

[29] A. V. Vasilakos, Z. Li, G. Simon, and W. You. Information centric network: Research challenges and opportunities. *J. Network and Computer Applications*, 52:1–10, 2015.

[30] A. Venkataramani, J. F. Kurose, D. Raychaudhuri, K. Nagaraja, M. Mao, and S. Banerjee. Mobilityfirst: a mobility-centric and trustworthy internet architecture. *Computer Communication Review*, 44(3):74–80, 2014.

[31] M. Walfish, H. Balakrishnan, and S. Shenker. Untangling the web from dns. In *Proceedings of the 1st Conference on Symposium on Networked Systems Design and Implementation - Volume 1*, NSDI '04, pages 17–17. USENIX Association, 2004.

[32] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos. A Survey of Information-Centric Networking Research. *IEEE Communications Surveys & Tutorials*, 16(2):1024–1049, 2013.

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

APPENDIX

Simulation hypothesis

- $\# \ IP = 3.5 * 10^9$, including mobile, estimate end 2016
- $\# \ AS = 7 * 10^4$, approx. 70.000 AS estimate end 2016
- $\# \ AS = \# \ CNS$, one Authoritative CNS per AS
- $\# \ \frac{IP}{AS} = 5 * 10^4$, approx. 50.000 IP "hosted" by each AS
- $\# \ iHN = 7 * 10^9$, approx. immutable, non unique, non proprietary HN[7]
- $\# \ mHN = 7 * 10^9$, approx. mutable, unique, proprietary HN[8]
- Def. Immutable ressources can be copied, held, and published by different IP (e.g. a movie, a song, a book), while mutable ressources cannot be copied, and can be hold and published by a single unique IP (e.g. a file Word, Excel)
- FAQ: Why $iHN = mHN$? It is just by chance ... because of the given hypothesis in footnotes ... modify the hypothesis for different values
- $HN = iHN \cup mHN$
- $\# \ HN = 1.4 * 10^{10}$
- NB. Each CNS querying for an HN and discovering it in another CNS, republish locally the HN.
- Hyp. Suppose a duplication rate factor of

  $$\{1, 5, 10, 50, 100, 1000, 10000, 100000\}\texttt{x}$$

  In one limit case, only one CNS holds a single HN and no other CNS is querying and republishing (ex: a local file queried only by IP in the same AS), while in the other limit case all the HN are queried by all CNS and byproduct republished in all CNS: the latter case forces to fix the size of the CNS memory, just like a cache (ex: the last Madonna CD queried worldwide and put in all the CNS memories by dropping another less demanded published HN, exactly like in Kademlia).
  - $HN = 1.4 * 10^{10}$
  - $fivHN = 7 * 10^{10}$
  - $tenHN = 1.4 * 10^{11}$
  - $ftyHN = 7 * 10^{11}$
  - $hunHN = 1.4 * 10^{12}$
  - $thoHN = 1.4 * 10^{13}$
  - $tthHN = 1.4 * 10^{14}$
  - $hthHN = 1.4 * 10^{15}$
- Def. We define two kind of memory: a fast nosql memory and a very fast cache memory. The nosql memorize all the publications of the current CNS, while the cache would be filled with the most requested publications, following a strategy like the one of e.g. Kademlia
- Publication of (republished) HN per CNS
  - $\# \ \frac{HN}{CNS} = 2 * 10^5$ (i.e. the size of a decent cache)
  - $\# \ \frac{fivHN}{CNS} = 5 * 10^5$
  - $\# \ \frac{tenHN}{CNS} = 2 * 10^6$
  - $\# \ \frac{ftyHN}{CNS} = 7 * 10^6$
  - $\# \ \frac{hunHN}{CNS} = 2 * 10^7$
  - $\# \ \frac{thoHN}{CNS} = 7 * 10^7$
  - $\# \ \frac{tthHN}{CNS} = 2 * 10^8$
  - $\# \ \frac{hthHN}{CNS} = 7 * 10^8$ (i.e. 100x the size of the Piratebay's nosql as in 2016)

[7]Ex: approx. 1000x the # of torrents in Piratebay's db, overestimation end 2016.

[8]Ex: approx. 2 personal files per IP.

Experiment 1: Bootstrap. We see the impact of varying α, β, γ in the distributed LINKS algorithm. All memories are empty. We measure the number of routing hops.
IN PROGRESS

Experiment 2: Rare content diffusion. A node query some unique rare content: if it found then it republish in the local CNS, else it fails the lookup. We measure the impact of republication of founded contents causing, byproduct, the shorten of the LINKS routing (success and failure query).
TO DO

Experiment 3: Flash crowd no cache. All the nodes queries an unique content: as usual, if they found then they republish on the local CNS, else they fail lookup. We measure the impact of republication of the founded content causing, byproduct, the shorten of the LINKS routing (success and failure query).
TO DO

Experiment 4: Flash crowd with cache. Still, all the nodes queries an unique context: we query (through the authoritative CNS) the "local nosql" base and the "cache" base via the LINKS routing (i.e. split instruction `lookupdb(HN)` in local-call and network-call). We measure the impact of the (fixed size) cache, whose politics is *most recent first*, in shortening the LINKS routing (success and failure query).
TO DO