Internet Engineering Task Force
Internet-Draft
Intended status: Experimental
Expires: July 30, 2015

M. Gallo
Alcatel-Lucent Bell Labs / IRT SystemX
D. Perino
Alcatel-Lucent Bell Labs
L. Muscariello
Orange / IRT SystemX
January 26, 2015

Content-Centric Networking Packet Header Format
draft-ccn-packet-header-00

Abstract

This document describes an experimental header format for CCN packets. The header is composed of a set fixed size fields followed by a set of Type-Length-Value fields in order to define a flexible, compact and easy to parse packet format for the CCN protocol.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Content-Centric Networking (CCN), initially proposed by PARC [CCN], is one of the architectures that were designed in the broad context of Information-Centric Networking (ICN). CCN, as more generally any ICN proposal, is centered around named content objects in contrast with today's host-centric nature of the TCP/IP architecture. Hence CCN packets carry a name that univocally represents a content object (or a segment of it) available in the network.

In CCN, a content object is represented by a unique hierarchical name. An entire content object, for example a file or a video, is composed of a set of segments, that are identified in the content object's name using a segment identifier (See Section 5.2).

Any client that needs to retrieve a content object or a specific content object's segment expresses a request (also referred to as Interest) specifying the name of the desired segment (also referred to as Data). Notice that a request an a content object or prefix name may be used by the CCN protocol for discovery mechanisms, not

discussed in this memo. The request is forwarded by CCN nodes or routers towards the potential source(s) that can serve it. CCN nodes can be equipped with content stores (local caches) from which they can serve the request. Hence, before propagating (forwarding) an Interest packet, a CCN node checks if it has the content segment requested by the Interest packet, in its local content store. If yes, it sends back a Data packet containing the desired content segment, and refrain from forwarding the Interest packet. If it does not have the content segment, it forwards the Interest packet based on its name-based Forwarding table.

In order to populate the name-based forwarding tables, the CCN control plane distributes name prefixes (formed by any subset of components from the content names) to CCN routers in order to allow name based request routing towards potential data sources. In addition to the name-based forwarding tables, intermediate CCN nodes also keep track of pending requests: forwarded requests that are still waiting for an answer. This allows them to send back the required Data when they receive it from upstream nodes. Besides, requests with exactly the same name may be aggregated and not forwarded upstream.

One of the most popular implementation for such networks is the CCNx prototype [CCNx]. CCNx has been initially proposed with a binary XML encoding which limits the possibility of implementing CCN over high speed equipments. This draft proposes a compact and flexible CCN packet header format in order to reduce the parsing delay needed by the XML encoding. Note that the packet header format that we define only specifies the fields that must be parsed by every CCN node (Core, Edge routers or end user machine).

As mentioned above, there are two main types of CCN packets, Interest packets that carry requests for content objects, and Data packets that carry the objects themselves. We next give an overview of these two types of packets

2. Content Centric Networking packets

Our CCN packet format includes: a fixed-size header (whose size depends on the packet type), a variable length content object's name, the optional name's TLVs (Section 5.1 , Section 5.2), some optional TLVs and finally the payload in some cases.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                                                               /
/                      Fixed size header                        /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Msg type           |           Msg Length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                    Variable Length  Name                      /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                     Optional Name TLVs                        /
/   ( Name Components Offsets TLV Name Segment ID Offset TLV)   /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                       Optional TLVs                           /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                                                               /
/                           Payload                             /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The only difference between Interest and Data packets is in the TLVs that are inserted just after the fixed size header.

Packet Type code	Packet Type	Packet Description
0x00	Interest packet.	Interest packet, used to express a request on a specific Content Object.
0x01	Data packet.	Data packet, used to satisfy a request expressed by an Interest.

Table 1: pkt types

The versioning field is one type and used to combine different protocol versions. The pkt type and the pkt length fields are

respectively 1 and 2 Bytes each. The motivation behind this choice is twofold:

0 In principle, the number of packet types needed in the CCN protocol should be small;

0 With a pkt length expressed with 2 Bytes, the maximum Length of a CCN packet is limited to 65 KBytes (including the packet header).

With a maximum packet size of 65 KBytes, CCN packets may not meet the Maximum Transmission Unit (MTU) of the underlying communication protocol (e.g. Ethernet MTU is 1500 Bytes). In this case, CCN packets must be fragmented and reassembled on a hop-by-hop basis, avoiding to break the one Interest / one Data correspondence of the CCN protocol. Fragmentation and reassembly are out of the scope of this draft, and can be addressed at a common convergence layer. The convergence layer may also be used in order to transfer topological information as Hop Counter, etc. However, such convergence layer is not described in this document.

After the pkt type and length fields that define the packet type (Interest or Data) and its length, there is one byte Hop Limit used to limit the scope of the packet. After the hop limit, there is a Reserved/Flags field. This field is not defined yet but can be used to host some useful flags and possibly to extend the fixed size header (i.e. a field that must be inserted in the fixed size header may be optional and its presence can be indicated by a special flag).

The Header Length field is similar to the Pkt Length one but it indicates the length of the Hop-by-Hop header. Additional fields may be introduced in the Hop-by-Hop header, especially for Hop-by-Hop packet frgamentation and reassebly. Such fields are not yet defined but could be introduced in a Type Lenth Value format.

After fixed size header, there are the Msg type and Msg Length fields of 2 Bytes each. Their meaning is similar to the Pkt Type and Pkt Length. The reason why Msg. type and Msg. Length are inserted after the fixed size header is that, in case of fragmentation and reassembly, once the fixed headers are removed, the reconstructed CCN message must be self describing.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    Version    |  Pkt  Type    |         Pkt Length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |   Hop Limit   | Flags/Reserved|            Hdr Length         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

2.1. Interest Packets

Interest packets are composed of the common fixed-size header plus the Msg. Type and Msg. length, repeated in ordrer to identify the message once reconstructed in case of Hop-by-Hop fragmentation and reassembly.

Note that with the defined Interest fixed-size header, the Name will always start at the 13th Byte of the Interest packet header.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    Version    |     0x00      |         Pkt Length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |   Hop Limit   | Flags/Reserved|            Hdr Length         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            Msg type           |            Msg Length         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   /                     Variable Length  Name                     /
   /                                                               /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   /                      Optional Name TLVs                       /
   /   ( Name Components Offsets TLV Name Segment ID Offset TLV)   /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   /                     Optional Interest TLVs                    /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

2.2. Data Packets

The Data packet header is very similar to the interest one.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    Version    |     0x01      |         Pkt Length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |   Hop Limit   | Flags/Reserved|            Hdr Length         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            Msg type           |            Msg Length         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   /                    Variable Length  Name                      /
   /                                                               /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   /                     Optional Name TLVs                        /
   /   ( Name Components Offsets TLV Name Segment ID Offset TLV)   /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   /                     Optional Data TLVs                        /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   /                                                               /
   /                           Payload                             /
   /                                                               /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3. Type-Length-Value formats

In order to encode variable length fields in CCN packets, we choose a single TLV encoding for simplicity composed by 2Bytes Type and 2Bytes Length. Despite this experimental choice, additional TLV types (i.e., 1B+2B, 1B+1B) may be needed and defined in the future using the most significant bits of the Type field.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                Type           |             Length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Length field represents the size of the field Value expressed in Bytes, excluding the preceding Type and Length fields.

With the above described TLV format up to 16K+ different fields can be defined with a maximum length of 65 KBytes each.

4. TLV Types

In the following a list of all the supported TLV types in the current packet format.

First Byte	Field Name	Field Description
0x00	Compact Name Components' Offset	Offsets (expressed using 1 Byte) of the components in the Name field.
0x01	Extended Name Components' Offset	Offsets (expressed using 2 Bytes) of the components in the Name field.
0x02	Name Segment ID's Offset	Offset of the segment ID in the Name field.
0x04	Interest Lifetime	Validity time for an Interest expressed in milliseconds.
0x05	Data Lifetime	Validity time for a Data Payload expressed in seconds.
0x06	Signature	Author's Signature for the Data payload
0x07	Key	The Content Object(s) that should be requested in order to retrieve the key used to sign the Data packet.

Table 2: TLV Types

5. Name encoding

In CCN, Names are assumed to be a hierarchical list of components separated by a mandatory predefined separator (i.e. "/" as default) similarly to URIs [RFC3986]. Hierarchical Content Object's names allow the network to group names with a common prefix in a single entry needed for name based request forwarding, hence significantly reducing the Name-Based forwarding tables' size.

Obviously, Inside a CCN packet header the Content Object's name is mandatory and must be accessible in a fast way by routers that simply need to read the name and forward the packet according to it. For that reason, the name is always put in a fixed location inside the packet header. Thanks to its fixed location (See Section 2.1 for Interest and Section 2.2 for Data) CCN names does not need to be encoded with a dedicated TLV. Right after the packet type parsing (Interest or Data), the CCN router exactly knows where the name is located.

The Name information in the CCN packet header is represented by two mandatory and two optional fields:

0 A mandatory Length field of 2 Bytes that defines the size of the name expressed in Bytes;

0 A mandatory Field of variable size representing the CCN hierarchical name;

0 An optional Name Components Offsets TLV (Section 5.1) that immediately follows the Name field. The Name Components Offsets TLV can be omitted in case of names composed by a single component;

0 An optional Name Segment Offset TLV (Section 5.2) that immediately follows the Name Components Offsets TLV. The Name Segment Offset TLV can be omitted in case of names without Segment identifier.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Name Length          |                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               /
/                  Variable Length  Name                        /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                      Optional Name TLVs                       /
/   ( Name Components Offsets TLV Name Segment ID Offset TLV)   /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Notice that the name can be logically divided in two parts: (1) the name identifying the complete content object (a collection of segments like a file or a video) ; it is delimited by the last component offset and (2) the segment identifier that together with the content object name uniquely identify a content object's segment in the network. Application specific components or names are not part of the CCN protocol layer and can be added as separate optional TLVs.

5.1. Name Components Offsets TLV format

The Name Components Offset TLV specifies the offsets that have to be applied to the Name field in order to retrieve the name's components. There are two different types of the Name Components Offsets TLV, one compact and one extended that are identified as two separate Type codes but that have the same meaning; only one type between the two can be specified per packet.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           0x01                |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-++-+-+-+-+-+-+-+-+/
/             Variable Length Name Component Offsets            /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Value field of this TLV has a fixed structure. In its compact version, (Type 0x00 and 0x40) it consists of a list of 1 Bytes fields that specify the offsets in the Name field of the different components. The first offset value indicates the name length up to the first component (component separator excluded), the second one the Name length up to the second component and so on for the other offsets. The last component delimits the content object/prefix identifier: this value corresponds to the end of the name if the packet does not have a Name Segment ID Offset TLV, or to the Name Segment Offset TLV value if present. In the extended name components offset TLV version (Type code 0x01 or 0x41), the name component's offset are represented using a list of 2 Bytes fields (instead of 1 Bytes fields).

```
 0
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|Compact Offset |
+-+-+-+-+-+-+-+-+
```

```
 0                   1
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Extended Offset        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

According to the Type of Name Components Offsets TLV (compact or extended), the number of name components can be immediately obtained by dividing by one (compact) or two (extended) the Name Components Offsets TLV's Length field.

The Name Components Offsets TLV can be omitted if the name has a single component but, if present, it has to be put immediately after the Name. If the offset TLV is not present, the parser assumes a name composed by a single component and without any segment identifier.

5.2. Name Segment ID Offset TLV format

The Name Segment ID Offset TLV, is used to specify the Segment ID offset. Contrarily to the Name components that indicates the end of a component field, the Name Segment ID Offset TLV indicates the starting point of the Segment ID inside the Name field. The Value field is composed by one offset represented using the offset extended version (2 Bytes).

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      0x02                     |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Segment ID Offset       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The Segment ID is a special name component and is assumed to be the last, and its value corresponds to the last component offset. The segment ID has a separate offset with respect to other components for essentially 2 reasons:

0 It can be used to explicitly differentiate the content object's name from the segment one (content object name plus Segment ID);

0 It can be used by some specific caching techniques (i.e. redundancy elimination) that are not described in this memo.

Moreover, the Name Segment Offset TLV only exists when the name has more than one component and the Name Components Offsets TLV is present in the packet header. The Name Segment Offset TLV if

present, must appear immediately after the Name Components Offsets TLV.

6. Interest TLVs

Interest packets may have additional TLVs that, if present must be parsed by any CCN Node as they may affect the name based request forwarding operation. Those fields are introduced as optional TLVs.

6.1. Interest Lifetime TLV format

Interest Lifetime is the time expressed in milliseconds for which the Interest remains valid. This means that CCN Nodes must maintain the state needed to re-route back the Data packets on the downstream at least for this amount of time. Each CCN Node can have a default Lifetime and ignore the one indicated in the Interest packet if it is not compliant with its own policy

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           0x04                |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-++-+-+-+-+-+-+-+-+/
/                        Interest Lifetime                      /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

7. Data TLVs

Data packets may have additional TLVs that, if present must be parsed by any CCN Node as they may affect the packet processing operation. Those fields are introduced as optional TLVs.

7.1. Data Lifetime TLV format

The Data Lifetime TLV is very similar to the Interest Lifetime TLV (See Section 6.1). It indicates the validity of a Content Objects expressed in seconds. After that time, the Data must be considered as not valid and discarded (or ignored) if stored in the local cache.

As for the Interest lifetime, the Data lifetime can be ignored and set to a default value if it is not compliant with the router policy.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           0x05                |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-++-+-+-+-+-+-+-+-+/
/                           Data Lifetime                       /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

7.2. Data Signature TLV format

The Signature TLV is the field used by the Content Object's publisher to digitally sign the Data he/she published. Any node in the network can verify the Content Object's authenticity by using the public Key of the Publisher that must be provided in a separate TLV.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           0x06                |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-++-+-+-+-+-+-+-+-+/
/                             Signature                         /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

7.3. Data Signature's Key TLV format

The Key TLV is basically the name under which the Publisher's Public key is available. Its value is used to retrieve the Public Key (and related information) of the authority to verify the signature. This TLV must be present if the Signature TLV is used in a Data packet.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           0x07                |            Length             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-++-+-+-+-+-+-+-+-+/
   /                      Signature's Key                          /
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

8. Acknowledgment

9. Security Considerations

This memo raises no security issues; however, according to [RFC2223], your document should contain a section near the end that discusses the security considerations of the protocol or procedures that are the main topic of your document.

10. References

[CCN] Jacobson, V., Smetters, D., Thorton, J., Plass, M., Briggs, N., and R. Braynard, "Networking Named Content", In proc. of IEEE CoNEXT, 2009.

[CCNx] Palo Alto Research Center (PARC), "CCNx Project", 2007, <http://www.ccnx.org>.

[RFC2223] Postel, J. and J. Reynolds, "Instructions to RFC Authors", RFC 2223, October 1997.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

Authors' Addresses

Massimo Gallo
Alcatel-Lucent Bell Labs / IRT SystemX

EMail: massimo.gallo@alcatel-lucent.com, massimo.gallo@irt-systemx.fr

Diego Perino
Alcatel-Lucent Bell Labs

EMail: diego.perino@alcatel-lucent.com

Luca Muscariello
Orange / IRT SystemX

EMail: luca.muscariello@orange.com, luca.muscariello@irt-systemx.fr