ICN Research Group
Internet-Draft
Intended status: Experimental
Expires: September 12, 2019

H. Asaeda
A. Ooka
NICT
X. Shao
Kitami Institute of Technology

March 11, 2019

# CCNinfo: Discovering Content and Network Information in Content-Centric Networks

draft-irtf-icnrg-ccninfo-01

## Abstract

This document describes a mechanism named "CCNinfo" that discovers information about the network topology and in-network cache in Content-Centric Networks (CCN). CCNinfo investigates: 1) the CCN routing path information per name prefix, 2) the Round-Trip Time (RTT) between content forwarder and consumer, and 3) the states of in-network cache per name prefix.

## Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 12, 2019.

## Copyright Notice

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

## 1. Introduction

In Content-Centric Networks (CCN), publishers provide content through the network, and receivers retrieve content by name. In this network architecture, routers forward content requests by means of their Forwarding Information Bases (FIBs), which are populated by name-based routing protocols. CCN also enables receivers to retrieve content from an in-network cache.

In CCN, while consumers do not generally need to know which content forwarder is transmitting the content to them, operators and developers may want to identify the content forwarder and observe the routing path information per name prefix for troubleshooting or investigating the network conditions.

Traceroute [6] is a useful tool for discovering the routing conditions in IP networks as it provides intermediate router

addresses along the path between source and destination and the Round-Trip Time (RTT) for the path. However, this IP-based network tool cannot trace the name prefix paths used in CCN. Moreover, such IP-based network tool does not obtain the states of the in-network cache to be discovered.

This document describes the specification of "CCNinfo", an active networking tool for discovering the path and content caching information in CCN. CCNinfo is designed based on the work previously published in [5].





CCNinfo can be implemented with the ccninfo user command and the forwarding function implementation on a content forwarder (e.g., router). The CCNinfo user (e.g., consumer) invokes the ccninfo command (described in Appendix A) with the name prefix of the content. The ccninfo command initiates the "Request" message (described in Section 3.1). The Request message, for example, obtains routing path and cache information. When an appropriate adjacent neighbor router receives the Request message, it retrieves cache information. If the router is not the content forwarder for the request, it inserts its "Report" block (described in Section 3.1.2) into the Request message and forwards the Request message to its upstream neighbor router(s) decided by its FIB. These two message types, Request and Reply messages, are encoded in the CCNx TLV format [1].

In this way, the Request message is forwarded by routers toward the content publisher, and the Report record is inserted by each intermediate router. When the Request message reaches the content forwarder (i.e., a router who can forward the specified cache or content), the content forwarder forms the "Reply" message (described in Section 3.2) and sends it to the downstream neighbor router. The Reply message is forwarded back toward the user in a hop-by-hop manner. This request-reply message flow, walking up the tree from a consumer toward a publisher, is similar to the behavior of the IP multicast traceroute facility [7].

CCNinfo supports multipath forwarding. The Request messages can be forwarded to multiple neighbor routers. When the Request messages forwarded to multiple routers, the different Reply messages will be

forwarded from different routers or publisher.





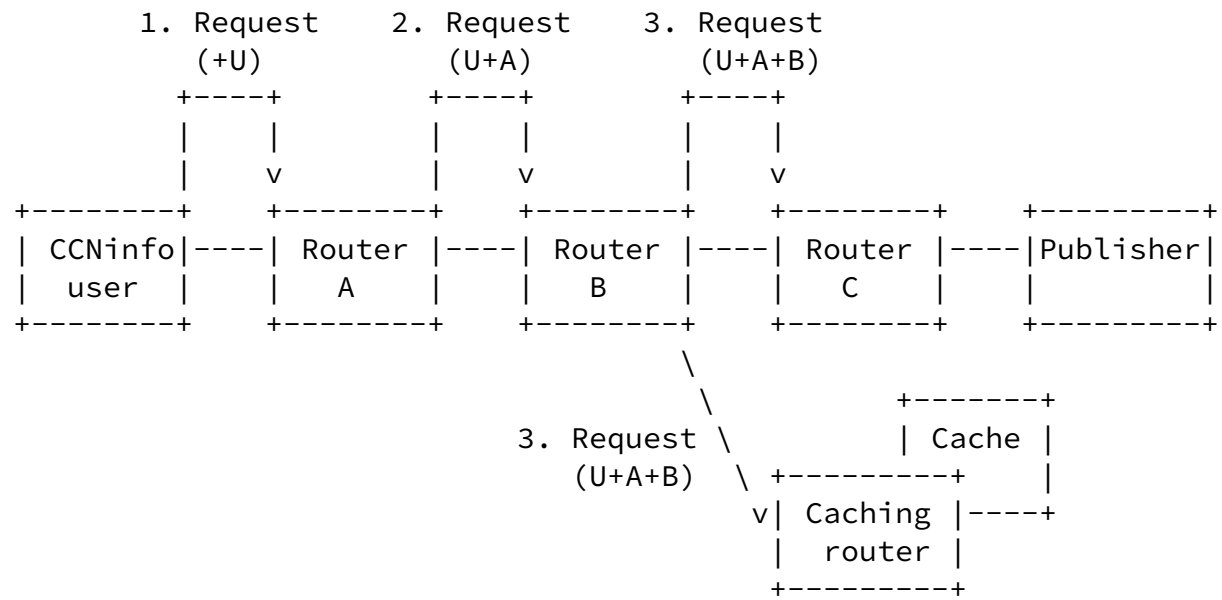


Figure 1: Request messages forwarded by consumer and routers. CCNinfo user and routers (i.e., Router A,B,C) insert their own Report blocks into the Request message and forward the message toward the content forwarder (i.e., caching router and publisher)

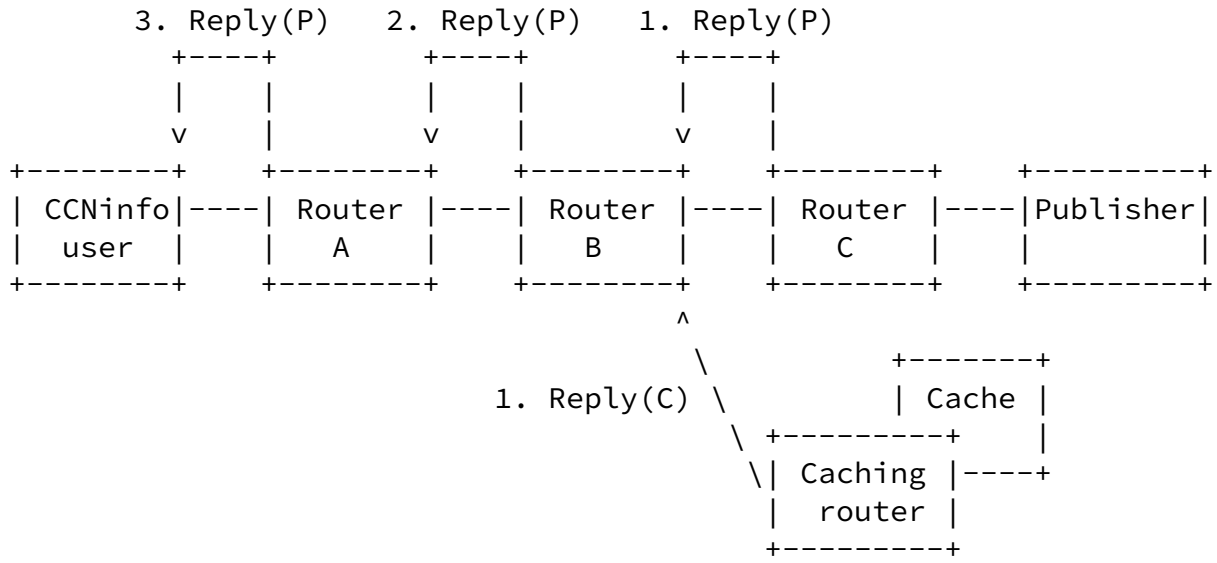


Figure 2: Default behavior. Reply messages forwarded by routers. Each router forwards the Reply message along its PIT entry, and finally the CCNinfo user receives a Reply message from Router C, which is the first-hop router for Publisher. Another Reply message from Caching router is discarded at Router B as the corresponding Reply message was already forwarded.





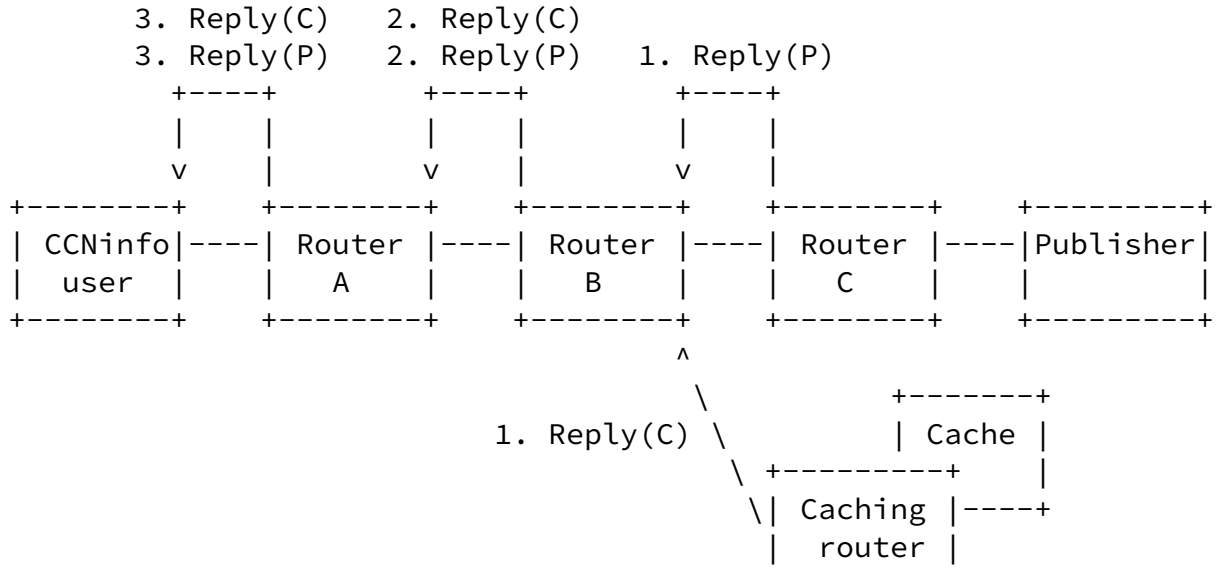

Figure 3: Full discovery request. Reply messages forwarded by publisher and routers. Each router forwards the Reply message along its PIT entry, and finally the CCNinfo user receives two Reply messages: one from the first-hop router and the other from the caching router.

CCNinfo facilitates the tracing of a routing path and provides: 1) the RTT between content forwarder (i.e., caching router or first-hop router) and consumer, 2) the states of in-network cache per name prefix, and 3) the routing path information per name prefix.

In addition, CCNinfo identifies the states of the cache, such as the following metrics for Content Store (CS) in the content forwarder: 1) size of the cached content objects, 2) number of the cached content objects, 3) number of the accesses (i.e., received Interests) per content, and 4) elapsed cache time and remain cache lifetime of content.

Furthermore, CCNinfo implements policy-based information provisioning that enables administrators to "hide" secure or private information, but does not disrupt the forwarding of messages. This policy-based information provisioning reduces the deployment barrier faced by operators in installing and running CCNinfo on their routers.

## 2. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in RFC 2119 [3], and indicate requirement levels for compliant CCNinfo implementations.





### 2.1. Definitions

Since CCNinfo requests flow in the opposite direction to the data flow, we refer to "upstream" and "downstream" with respect to data, unless explicitly specified.

Router
: It is a router facilitating CCN-based content retrieval in the

path between consumer and publisher.

Scheme name
: It indicates a URI and protocol. This document only considers "ccn:/" as the scheme name.

Prefix name
: A prefix name, which is defined in [2], is a name that does not uniquely identify a single content object, but rather a namespace or prefix of an existing content object name.

Exact name
: An exact name, which is defined in [2], is one which uniquely identifies the name of a content object.

Node
: It is a router, publisher, or consumer.

Content forwarder
: It is either a caching router or a first-hop router that forwards content objects to consumers.

CCNinfo user
: It is a node that invokes the ccninfo command and initiates the CCNinfo Request.

Incoming face
: The face on which data is expected to arrive from the specified name prefix.

Outgoing face
: The face to which data from the publisher or router is expected to transmit for the specified name prefix. It is also the face on which the Request messages are received.

Upstream router
: The router, connecting to the Incoming face of a router, which is responsible for forwarding data for the specified name prefix to the router.





First-hop router (FHR)

The router that is directly connected to the publisher.

Last-hop router (LHR)
The router that is directly connected to the consumers.

## 3. CCNinfo Message Formats

CCNinfo uses two message types: Request and Reply. Both messages are encoded in the CCNx TLV format ([1], Figure 4). The Request message consists of a fixed header, Request block TLV Figure 8, and Report block TLV(s) Figure 11. The Reply message consists of a fixed header, Request block TLV, Report block TLV(s), and Reply block/sub-block TLV(s) Figure 14.

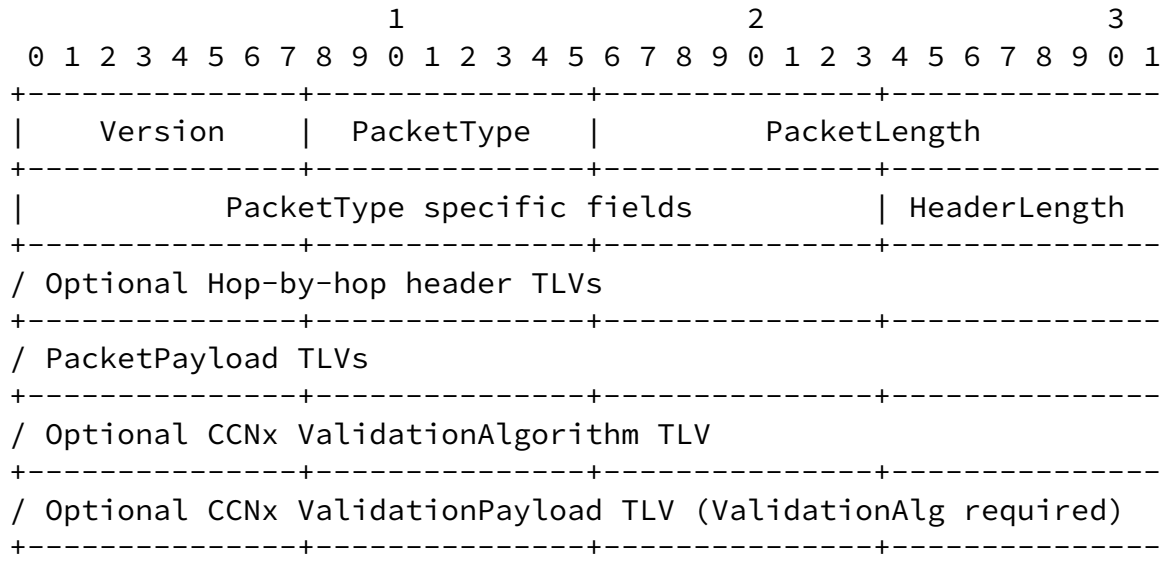


Figure 4: Packet format [1]

The Request and Reply Type values in the fixed header are PT_REQUEST and PT_REPLY, respectively (Figure 5). These messages are forwarded in a hop-by-hop manner. When the Request message reaches the content forwarder, the content forwarder turns the Request message into a Reply message by changing the Type field value in the fixed header from PT_REQUEST to PT_REPLY and forwards back to the node that has initiated the Request message.

| Code | Type name |
|---|---|
| %x00 | PT_INTEREST [1] |
| %x01 | PT_CONTENT [1] |
| %x02 | PT_RETURN [1] |
| %x03 | PT_REQUEST |
| %x04 | PT_REPLY |

Figure 5: Packet Type Namespace

The CCNinfo Request and Reply messages MUST begin with a fixed header with either a Request or Reply type value to specify whether it is a Request message or Reply message. Following a fixed header, there can be a sequence of optional hop-by-hop header TLV(s) for a Request message. In the case of a Request message, it is followed by a sequence of Report blocks, each from a router on the path toward the publisher or caching router.

At the beginning of PacketPayload TLVs, one top-level TLV type, T_DISCOVERY (Figure 6), exists at the outermost level of a CCNx protocol message. This TLV indicates that the Name segment TLV(s) and Reply block TLV(s) would follow in the Request or Reply message.

| Code | Type name |
|---|---|
| %x0000 | Reserved [1] |
| %x0001 | T_INTEREST [1] |
| %x0002 | T_OBJECT [1] |
| %x0003 | T_VALIDATION_ALG [1] |
| %x0004 | T_VALIDATION_PAYLOAD [1] |
| %x0005 | T_DISCOVERY |

Figure 6: Top-Level Type Namespace

## 3.1. Request Message

When a CCNinfo user initiates a discovery request (e.g., by ccninfo command described in Appendix A), a CCNinfo Request message is created and forwarded to its upstream router through the Incoming face(s) determined by its FIB.

The Request message format is as shown in Figure 7. It consists of a fixed header, Request block TLV (Figure 8), Report block TLV(s) (Figure 11), and Name TLV. The Type value of Top-Level type namespace is T_DISCOVERY (Figure 6). The Type value for the Report message is PT_REQUEST.

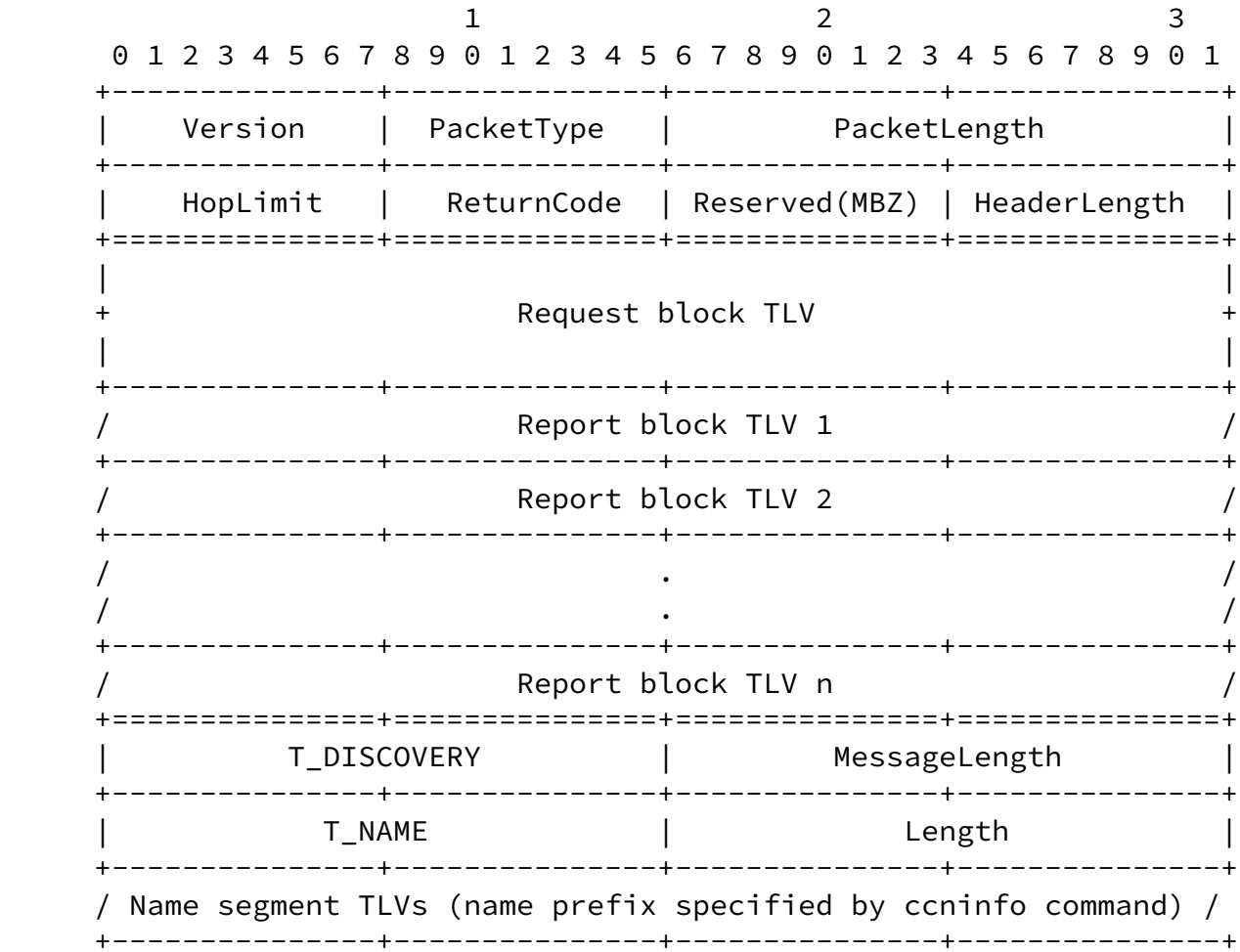


Figure 7: Request message consists of a fixed header, Request block TLV, Report block TLV(s), and Name TLV

HopLimit: 8 bits

HopLimit is a counter that is decremented with each hop whenever a Request packet is forwarded. It limits the distance a Request may travel on the network.

ReturnCode: 8 bits

ReturnCode is used for the Reply message. This value is replaced

by the content forwarder when the Request message is returned as the Reply message (see Section 3.2). Until then, this field MUST be transmitted as zeros and ignored on receipt.





| Value | Name | Description |
|---|---|---|
| %x00 | NO_ERROR | No error |
| %x01 | WRONG_IF | CCNinfo Request arrived on an interface to which this router would not forward for the specified name/function toward the publisher. |
| %x02 | INVALID_REQUEST | Invalid CCNinfo Request is received. |
| %x03 | NO_ROUTE | This router has no route for the name prefix and no way to determine a potential route. |
| %x04 | NO_INFO | This router has no cache information for the specified name prefix. |
| %x05 | NO_SPACE | There was not enough room to insert another Report block in the packet. |
| %x06 | INFO_HIDDEN | Information is hidden from this discovery because of some policy. |
| %x0E | ADMIN_PROHIB | CCNinfo Request is administratively prohibited. |
| %x0F | UNKNOWN_REQUEST | This router does not support/recognize the Request message. |
| %x80 | FATAL_ERROR | A fatal error is one where the router may know the upstream router but cannot forward the message to it. |

Reserved (MBZ): 8 bits

The reserved fields in the Value field MUST be transmitted as zeros and ignored on receipt.

### 3.1.1. Request Block

When a CCNinfo user transmits the Request message, it MUST insert the

Request block TLV (Figure 8) and the Report block TLV (Figure 11) of its own to the Request message before sending it through the Incoming face(s).





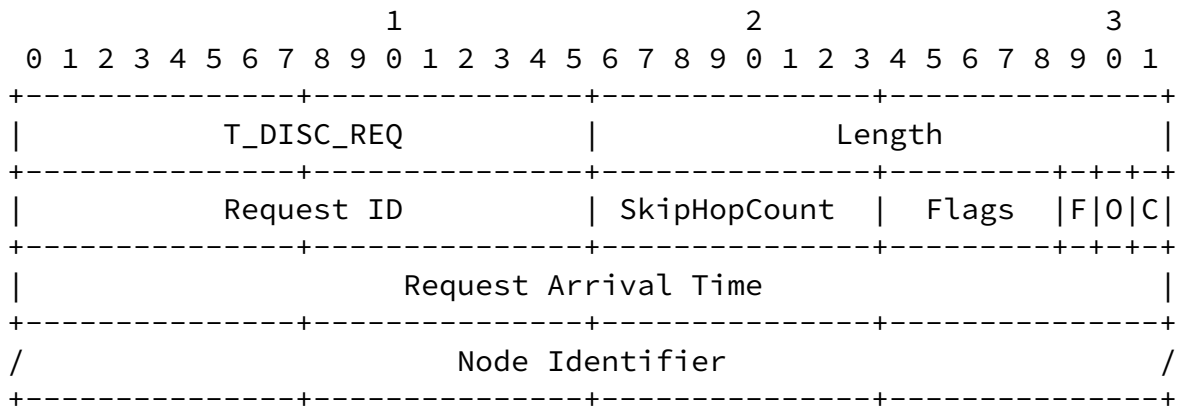


Figure 8: Request block TLV (hop-by-hop header)

| Code | Type name |
|---|---|
| %x0000 | Reserved [1] |
| %x0001 | T_INTLIFE [1] |
| %x0002 | T_CACHETIME [1] |
| %x0003 | T_MSGHASH [1] |
| %x0004-%x0007 | Reserved [1] |
| %x0008 | T_DISC_REQ |
| %x0009 | T_DISC_REPORT |
| %x0FFE | T_PAD [1] |

| | |
|---|---|
| %x0FFF | T_ORG [1] |
| %x1000-%x1FFF | Reserved [1] |

Figure 9: Hop-by-Hop Type Namespace

Type: 16 bits

Format of the Value field. For the single Request block TLV, the type value MUST be T_DISC_REQ. For all the available types for hop-by-hop type namespace, please see Figure 9.

Length: 16 bits

Length of Value field in octets.

Request ID: 16 bits

This field is used as a unique identifier for this CCNinfo Request so that duplicate or delayed Reply messages can be detected.

SkipHopCount: 8 bits





Number of skipped routers for a Request. This value MUST be lower than the value of HopLimit at the fixed header.

Flags: 16 bits

Flags field is used to indicate the types of the content or path discoveries. Currently, as shown in Figure 10, three bits, "C", "O", and "F", are assigned, and the other 5 bits are reserved (MBZ) for the future use. These flags are set by CCNinfo users when they initiate Requests (see Appendix A), and routers that receive the Requests deal with the flags and change the behaviors (see Section 5 for details).

| Flag | Value | Description |
|---|---|---|
| C | 0 | Path discovery (i.e., no cache information retried) |
| C | 1 | Cache information retrieval (default) |
| O | 0 | Request to any content forwarder (default) |

| | | |
|---|---|---|
| O | 1 | Publisher reachability (i.e., only FHR can reply) |
| F | 0 | Request based on FIB's strategy (default) |
| F | 1 | Full discovery request. Request to multiple upstream routers simultaneously |

Figure 10: Codes and types specified in Flags field

### 3.1.2. Report Block

A CCNinfo user and each upstream router along the path would insert its own Report block TLV without changing the Type field of the fixed header of the Request message until one of these routers is ready to send a Reply. In the Report block TLV (Figure 11), the Request Arrival Time and the Node Identifier MUST be inserted.

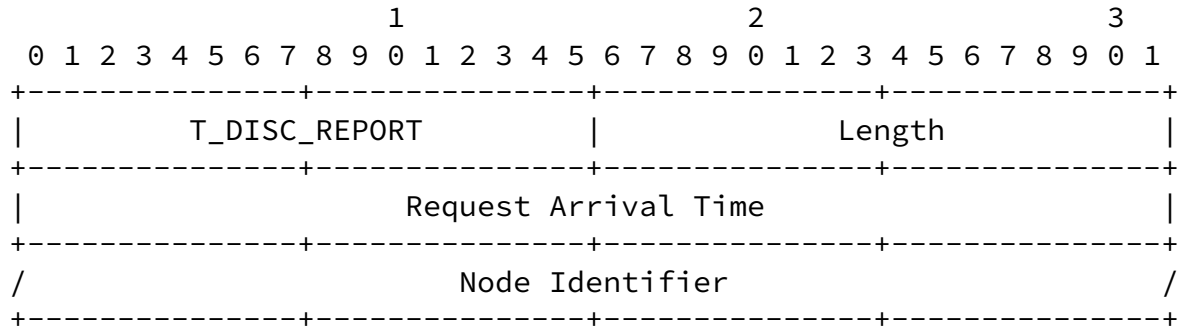


Figure 11: Report block TLV (hop-by-hop header)

Type: 16 bits

Format of the Value field. For the Report block TLV, the type value(s) MUST be T_DISC_REPORT in the current specification.

Length: 16 bits

Length of Value field in octets.

Request Arrival Time: 32 bits

The Request Arrival Time is a 32-bit NTP timestamp specifying the arrival time of the CCNinfo Request packet at this router. The

32-bit form of an NTP timestamp consists of the middle 32 bits of the full 64-bit form; that is, the low 16 bits of the integer part and the high 16 bits of the fractional part.

The following formula converts from a UNIX timeval to a 32-bit NTP timestamp:

```
request_arrival_time
= ((tv.tv_sec + 32384) << 16) + ((tv.tv_nsec << 7) / 1953125)
```

The constant 32384 is the number of seconds from Jan 1, 1900 to Jan 1, 1970 truncated to 16 bits. ((tv.tv_nsec << 7) / 1953125) is a reduction of ((tv.tv_nsec / 1000000000) << 16).

Note that it is RECOMMENDED that all the routers on the path to have synchronized clocks; however, if they do not have synchronized clocks, CCNinfo measures one-way latency.

Node Identifier: variable length

This field specifies the CCNinfo user or the router identifier (e.g., IPv4 address) of the Incoming face on which packets from the publisher are expected to arrive, or all-zeros if unknown or unnumbered. Since we may not always rely on the IP addressing architecture, it would be necessary to define the identifier uniqueness (e.g., by specifying the protocol family) for this field. However, defining such uniqueness is out of scope of this document. Potentially, this field may be defined as a new TLV based on the CCNx TLV format [1].

## 3.2. Reply Message

When a content forwarder receives a CCNinfo Request message from the appropriate adjacent neighbor router, it would insert a Reply block TLV and Reply sub-block TLV(s) of its own to the Request message and turn the Request into the Reply by changing the Type field of the fixed header of the Request message from PT_REQUEST to PT_REPLY. The





Reply message (see Figure 12) would then be forwarded back toward the CCNinfo user in a hop-by-hop manner.

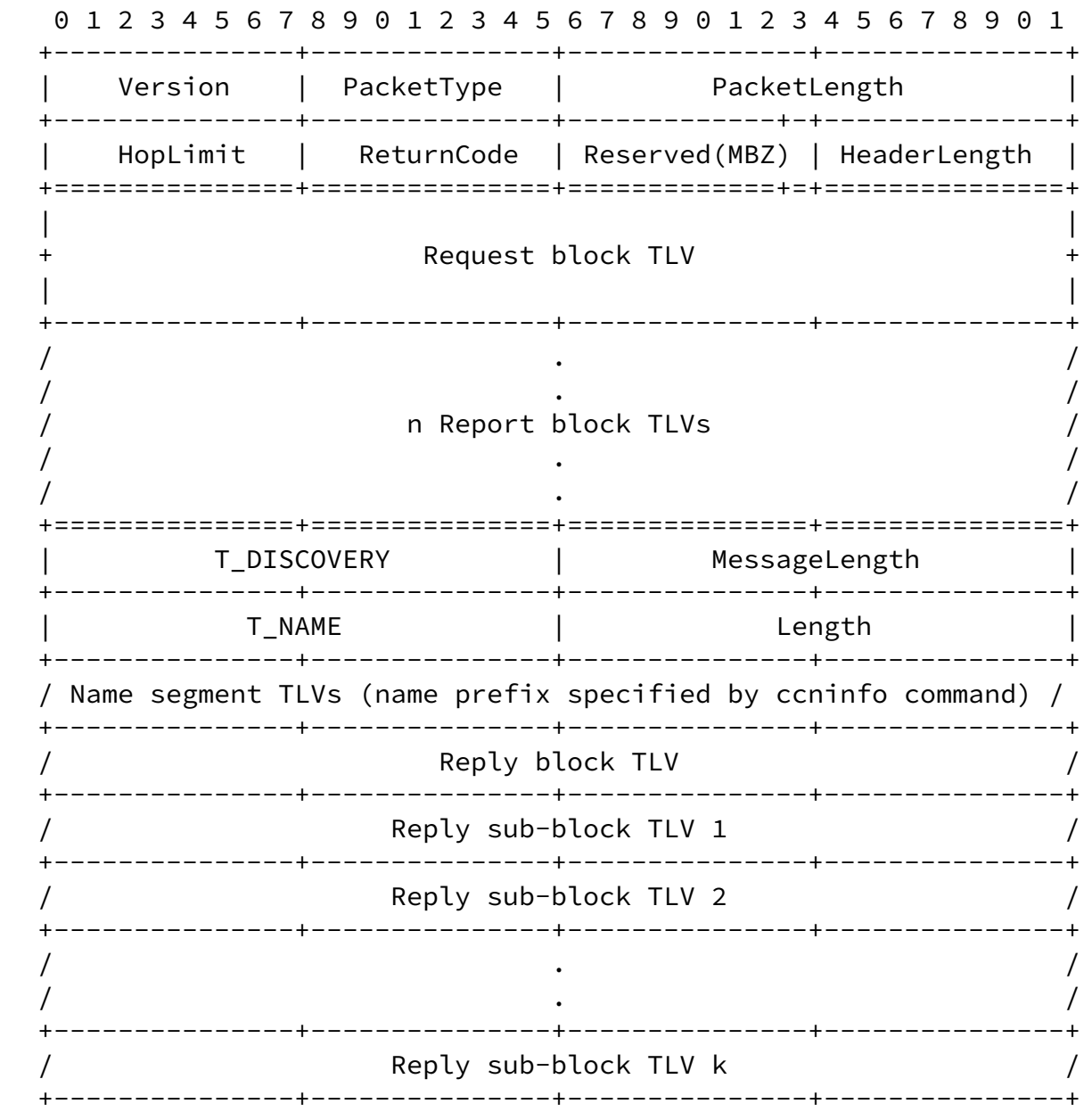


Figure 12: Reply message consists of a fixed header, Request block TLV, Report block TLV(s), Name TLV, and Reply block/sub-block TLV(s)

| Code | Type name |
|---|---|
| %x0000 | T_NAME [1] |
| %x0001 | T_PAYLOAD [1] |
| %x0002 | T_KEYIDRESTR [1] |
| %x0003 | T_OBJHASHRESTR [1] |
| %x0005 | T_PAYLDTYPE [1] |
| %x0006 | T_EXPIRY [1] |
| %x0007 | T_DISC_REPLY |
| %x0008-%x0012 | Reserved [1] |
| %x0FFE | T_PAD [1] |
| %x0FFF | T_ORG [1] |
| %x1000-%x1FFF | Reserved [1] |

Figure 13: CCNx Message Type Namespace

### 3.2.1. Reply Block

The Reply block TLV is an envelope for Reply sub-block TLV(s) (explained in Section 3.2.1.1).

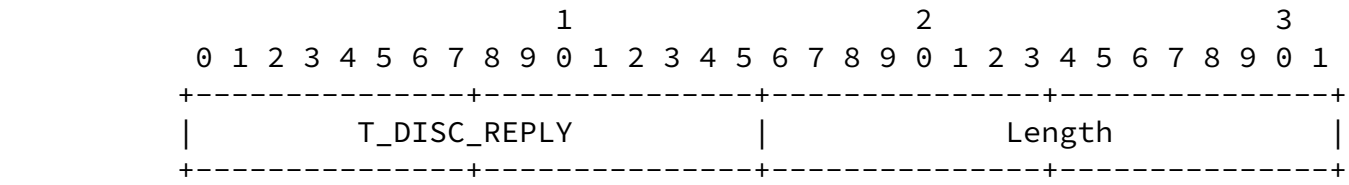


Figure 14: Reply block TLV (packet payload)

Type: 16 bits

Format of the Value field. For the Reply block TLV, the type value MUST be T_DISC_REPLY in the current specification.

Length: 16 bits

Length of Value field in octets. This length is a total length of Reply sub-block(s).

#### 3.2.1.1. Reply Sub-Block

In addition to the Reply block, a router on the traced path will add one or multiple Reply sub-blocks followed by the Reply block before sending the Reply to its neighbor router.

The Reply sub-block is flexible for various purposes. For instance, operators and developers may want to obtain various characteristics

of content such as content's ownership and copyright, or other cache





states and conditions. In this document, Reply sub-block TLVs for T_DISC_CONTENT and T_DISC_CONTENT_OWNER (Figure 15) are defined; other Reply sub-block TLVs will be defined in separate document(s).

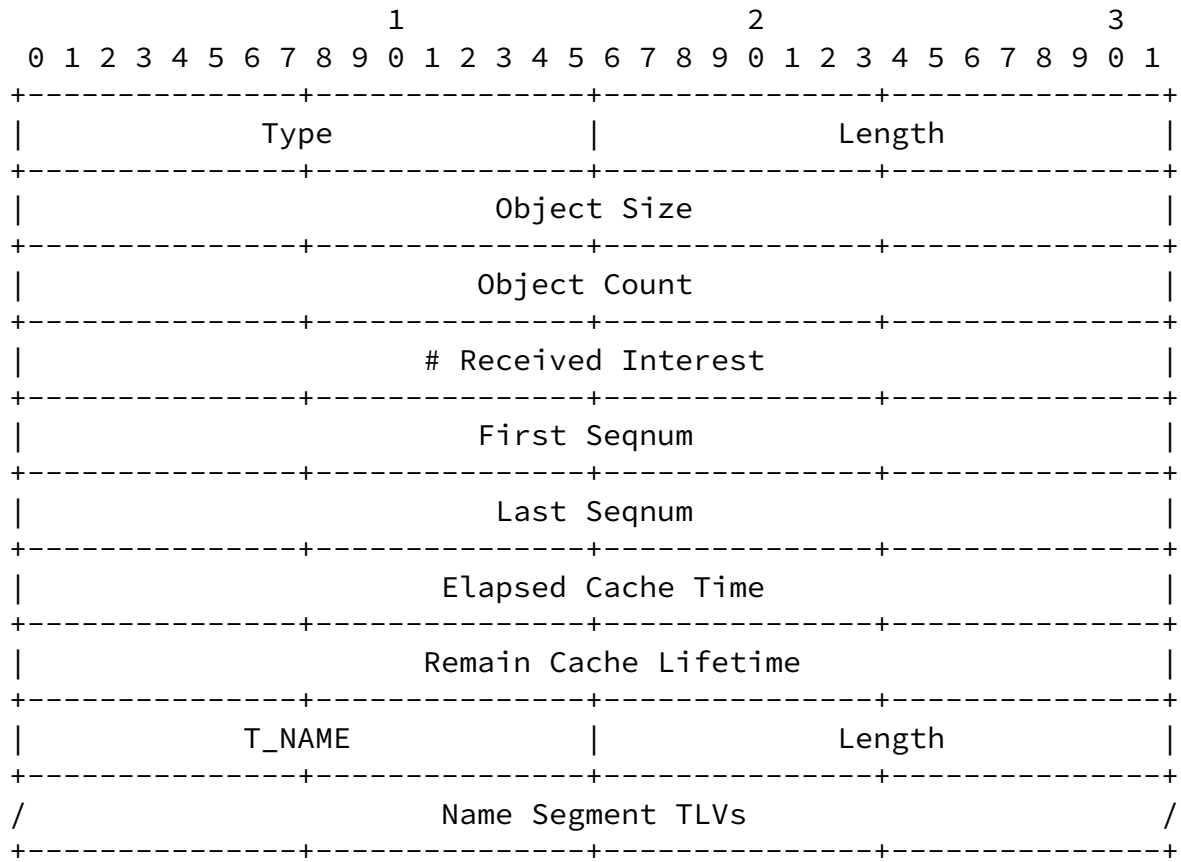


Figure 15: Reply sub-block TLV for T_DISC_CONTENT and T_DISC_CONTENT_OWNER (packet payload)

| Code | Type name |
|---|---|
| %x0000 | T_DISC_CONTENT |
| %x0001 | T_DISC_CONTENT_OWNER |
| %x0FFF | T_ORG |
| %x1000-%x1FFF | Reserved (Experimental Use) |

Figure 16: CCNinfo Reply Type Namespace

Type: 16 bits

Format of the Value field. For the Reply sub-block TLV, the type value MUST be one of the type value defined in the CCNinfo Reply Type Namespace (Figure 16). T_DISC_CONTENT is specified when the cache information is replied from a caching router. T_DISC_CONTENT_OWNER is specified when the content information is replied from a FHR attached to a publisher.





Length: 16 bits

Length of Value field in octets.

Object Size: 32 bits

The total size (byte) of the (cached) content objects. Note that the maximum size expressed by 32 bit field is about 4.29 GB. This value MAY be null when FHR sends the Reply message.

Object Count: 32 bits

The number of the (cached) content objects. Note that the maximum count expressed by 32 bit field is about 4.29 billion. This value MAY be null when FHR sends the Reply message.

# Received Interest: 32 bits

The total number of the received Interest messages to retrieve the cached content objects.

First Seqnum: 32 bits

The first sequential number of the (cached) content objects. This value MAY be null if the router does not know or cannot report.

Last Seqnum: 32 bits

The last sequential number of the (cached) content objects. Above First Seqnum and this Last Seqnum do not guarantee the consecutiveness of the cached content objects. This value MAY be null if the router does not know or cannot report.

Elapsed Cache Time: 32 bits

The elapsed time (seconds) after the oldest content object of the content is cached. This value MAY be null if the router does not know or cannot report.

Remain Cache Lifetime: 32 bits

The lifetime (seconds) of a content object, which is removed first among the cached content objects. This value MAY be null if the router does not know or cannot report.

Specification of the Name TLV (whose type value is T_NAME) and the Name Segment TLVs are described in [1], and CCNinfo follows that specification. CCNinfo also allows to specify the content name





either with a prefix name (such as "ccn:/news/today") or an exact name (such as "ccn:/news/today/Chunk=10"). When a CCNinfo user specifies a prefix name, s/he will obtain the information of the matched content objects in the content forwarder. On the other hand, when a CCNinfo user specifies an exact name, s/he will obtain only about the specified content object in the content forwarder.

# 4. CCNinfo User Behavior

## 4.1. Sending CCNinfo Request

The CCNinfo user's program (e.g., ccninfo command) enables user to obtain both the routing path information and in-network cache information in a same time.

A CCNinfo user initiates a CCNinfo Request by sending the Request message to the adjacent neighbor router(s) of interest. As a typical example, a CCNinfo user invokes the ccninfo command (detailed in Appendix A) that forms a Request message and sends it to the user's adjacent neighbor router(s).

When the CCNinfo user's program initiates a Request message, it MUST insert the necessary values, the "Request ID" (in the Request block) and the "Node Identifier" (in the Report block), in the Request and Report blocks. The Request ID MUST be unique for the CCNinfo user until s/he receives the corresponding Reply message(s) or times out

the Request.

Because of some policy, a router needs to validate CCNinfo Requests (whether it accepts the Request or not) especially when the router receives the "full discovery request" (see Section 5.3). To support this requirement, the CCNinfo user's program MAY require appending the user's signature into the CCNx ValidationPayload TLV. The router then forwards the Request message or reply the Reply message whenever it approves the Request.

After the CCNinfo user's program sends the Request message, until the Reply times out or the expected numbers of Replies or a Reply message having a non-zero ReturnCode in the fixed header is received, the CCNinfo user's program MUST keep the following information; Request ID and Flags specified in the Request block, Node Identifier and Request Arrival Time specified in the Report block, and HopLimit specified in the fixed header.





#### 4.1.1. Routing Path Information

A CCNinfo user can send a CCNinfo Request for investigating routing path information for the specified named content. By the Request, the legitimate user can obtain; 1) identifiers (e.g., IP addresses) of intermediate routers, 2) identifier of content forwarder, 3) number of hops between content forwarder and consumer, and 4) RTT between content forwarder and consumer, per name prefix. This CCNinfo Request is terminated when it reaches the content forwarder.

#### 4.1.2. In-Network Cache Information

A CCNinfo user can send a CCNinfo Request for investigating in-network cache information. By the Request, the legitimate user can obtain; 1) size of the cached content objects, 2) number of the cached content objects, 3) number of the accesses (i.e., received Interests) per content, and 4) lifetime and expiration time of the cached content object, for Content Store (CS) in the content forwarder. This CCNinfo Request is terminated when it reaches the

content forwarder.

### 4.2. Receiving CCNinfo Reply

A CCNinfo user's program will receive one or multiple CCNinfo Reply messages from the adjacent neighbor router that has previously received and forwarded the Request message(s). When the program receives the Reply, it MUST compare the kept Request ID and Node Identifier and the ones noted in the Request block TLV in the Reply. If they do not match, the Reply message MUST be silently discarded.

If the number of the Report blocks in the received Reply is more than the initial HopLimit value (which was inserted in the original Request), the Reply MUST be silently ignored.

After the CCNinfo user has determined that s/he has traced the whole path or as much as s/he can expect to, s/he might collect statistics by waiting a timeout. Useful statistics provided by CCNinfo can be seen in Section 8.

## 5. Router Behavior

### 5.1. User and Neighbor Verification

Upon receiving a CCNinfo Request message, a router MAY examine whether the message comes from a valid CCNinfo user. If the router recognizes that the Request sender's signature specified in the Request is invalid, it terminates the Request as defined in Section 6.3.





Upon receiving a CCNinfo Request/Reply message, a router MAY examine whether the message comes from a valid adjacent neighbor node. If the router recognizes that the Request/Reply sender is invalid, the Request/Reply message MUST be silently ignored. See Section 9.8.

### 5.2. Receiving CCNinfo Request

#### 5.2.1. Normal Processing

After the CCNinfo Request message verification, the router performs the following steps.

1. The value of the "HopLimit" in the fixed header and the value of the "SkipHopCount" in the Request block are counters that are decremented with each hop. If the HopLimit value is zero, the router terminates the Request as defined in Section 6.4. If the SkipHopCount value is equal or more than the HopLimit value, the router terminates the Request as defined in Section 6.3. Otherwise, until the SkipHopCount value becomes zero, the router forwards the Request message to the upstream router(s) without adding its own Report block and without replying the Request. If the router does not know the upstream router(s) for the specified name prefix, it terminates the Request as defined in Section 6.4.

2. The router examines the Flags field (specified in Figure 10) in the Request block of received CCNinfo Request. If the "C" flag is set but the "O" flag is not set, that is categorized as the "cache information discovery". If both the "C" and "O" flags are not set, that is categorized as the "routing path information discovery". If "O" flag is set, that is categorized as the "publisher discovery".

3. If the Request is either the "cache information discovery" or the "routing path information discovery", the router examines its FIB and CS. If the router caches the specified content, it inserts own Report block in the hop-by-hop header, and sends the Reply message with own Reply block and sub-block(s) (in case of cache information discovery) or sends the Reply message with own Reply block without adding any Reply sub-block (in case of routing path information discovery). If the router does not cache the specified content but knows the upstream neighbor router(s) for the specified name prefix, it inserts own Report block and forwards the Request to the upstream neighbor(s). If the router does not cache the specified content and does not know the upstream neighbor router(s) for the specified name prefix, it terminates the Request as defined in Section 6.4.





4. If the Request is the "publisher discovery", the router examines whether it is the FHR for the requested content. If it is the FHR, it sends the Reply message with own Report block and sub-block. If the router is not the FHR but knows the upstream neighbor router(s) for the specified name prefix, it adds the own

Report block and forwards the Request to the neighbor(s). If the node is not the FHR and does not know the upstream neighbor router(s) for the specified name prefix, it terminates the Request as defined in Section 6.4.

## 5.3. Forwarding CCNinfo Request

When a router decides to forward a Request message with its Report block to its upstream router(s), it specifies the Request Arrival Time and Node Identifier in the Report block of the Request message. The router then forwards the Request message upstream toward the publisher or caching router based on the FIB entry.

When the router forwards the Request message, it MUST record the Request ID, the F flag, and the Node Identifier specified in the Request block at the corresponding PIT entry. The router can later check the PIT entry to correctly forward back the Reply message(s). (See below.)

CCNinfo supports multipath forwarding. The Request messages can be forwarded to multiple neighbor routers. Some router may have strategy for multipath forwarding; when it sends Interest messages to multiple neighbor routers, it may delay or prioritize to send the message to the upstream routers. The CCNinfo Request, as the default, complies with such strategy; a CCNinfo user could trace the actual forwarding path based on the forwarding strategy.

On the other hand, there may be the case that a CCNinfo user wants to discover all potential forwarding paths based on routers' FIBs. The "full discovery request" enables this function. If a CCNinfo user sets the F flag in the Request block of the Request message (as seen in Figure 10) to request the full discovery, the upstream routers forward the Requests to the all multiple upstream routers based on the FIBs simultaneously. Then the CCNinfo user could trace the all potential forwarding paths. Note that some routers MAY ignore the full discovery request according to their policy. In that case, the router terminates the Request as defined in Section 6.10.

When the Request messages forwarded to multiple routers, the different Reply messages will be forwarded from different routers or publisher. To support this case, PIT entries initiated by CCNinfo remain until the configured CCNinfo Reply Timeout (Section 7.1) passes. In other words, unlike the ordinary Interest-Data

communications in CCN, the router SHOULD NOT remove the PIT entry created by the CCNinfo Request until the timeout value expires.

CCNinfo Requests SHOULD NOT result in PIT aggregation in routers during the Request message transmission.

## 5.4. Sending CCNinfo Reply

When a router decides to send a Reply message to its downstream neighbor router or the CCNinfo user with NO_ERROR return code, it inserts a Report block having the Request Arrival Time and Node Identifier to the hop-by-hop TLV header of the Request message. And then the router inserts the corresponding Reply block with an appropriate type value (Figure 15) and Reply sub-block(s) to the payload. The router does not insert any Reply block/sub-block if there is an error. The router finally changes the Type field in the fixed header from PT_REQUEST to PT_REPLY and forwards the message back as the Reply toward the CCNinfo user in a hop-by-hop manner.

If a router cannot continue the Request, it MUST put an appropriate ReturnCode in the Request message, change the Type field value in the fixed header from PT_REQUEST to PT_REPLY, and forward the Reply message back toward the CCNinfo user, to terminate the request. See Section 6.

## 5.5. Forwarding CCNinfo Reply

When a router receives a CCNinfo Reply whose Request ID and Node Identifier match the ones in the PIT entry and sent from a valid adjacent neighbor router, it forwards the CCNinfo Reply back toward the CCNinfo user. If the router does not receive the corresponding Reply within the [CCNinfo Reply Timeout] period, then it removes the corresponding PIT entry and terminates the trace.

Flags field in the Request block TLV is used to indicate whether the router keeps the PIT entry during the CCNinfo Reply Timeout even after one or more corresponding Reply messages are forwarded. When the CCNinfo user does not set the F flag (i.e., "0"), the intermediate routers immediately remove the PIT entry whenever they forward the corresponding Reply message. When the CCNinfo user sets the F flag (i.e., "1"), which means the CCNinfo user chooses the "full discovery request", the intermediate routers keep the PIT entry within the [CCNinfo Reply Timeout] period. After this timeout, the PIT entry is removed.

CCNinfo Replies MUST NOT be cached in routers upon the Reply message transmission.

# 6. CCNinfo Termination

When performing an expanding hop-by-hop trace, it is necessary to determine when to stop expanding. There are several cases an intermediate router might return a Reply before a Request reaches the caching router or the publisher.

## 6.1. Arriving at First-hop router

A CCNinfo Request can be determined to have arrived at the first-hop router.

## 6.2. Arriving at Router Having Cache

A CCNinfo Request can be determined to have arrived at the router having the specified content cache within the specified HopLimit.

## 6.3. Invalid Request

If the router does not accept the Request, the router MUST note a ReturnCode of INVALID_REQUEST in the fixed header of the message and forward the message without appending any Reply (sub-)block TLV as the Reply back to the CCNinfo user. The router MAY, however, randomly ignore the received invalid messages. (See Section 9.6.)

## 6.4. No Route

If the router cannot determine the routing paths or neighbor routers for the specified name prefix within the specified HopLimit, the router MUST note a ReturnCode of NO_ROUTE in the fixed header of the message and forward the message as the Reply back to the CCNinfo user.

## 6.5. No Information

If the router does not have any information about the specified name prefix within the specified HopLimit, the router MUST note a ReturnCode of NO_INFO in the fixed header of the message and forward the message as the Reply back to the CCNinfo user.

## 6.6. No Space

If appending the Report block would exceed the maximum (i.e., 255 byte) header length or make the CCNinfo Request message longer than the MTU of the Incoming face or longer than 1280 bytes (especially in the situation supporting IPv6 as the payload [4]), the router MUST note a ReturnCode of NO_SPACE in the fixed header of the message and forward the message as the Reply back to the CCNinfo user.





### 6.7. Fatal Error

A CCNinfo Request has encountered a fatal error if the last ReturnCode in the trace has the 0x80 bit set (see Section 3.1).

### 6.8. CCNinfo Reply Timeout

If a router receives the Request or Reply message that expires its own [CCNinfo Reply Timeout] value (Section 7.1), the router will silently discard the Request or Reply message.

### 6.9. Non-Supported Node

Cases will arise in which a router or a publisher along the path does not support CCNinfo. In such cases, a CCNinfo user and routers that forward the CCNinfo Request will time out the CCNinfo request.

### 6.10. Administratively Prohibited

If CCNinfo is administratively prohibited, the router rejects the Request message and MUST reply the CCNinfo Reply with the ReturnCode of ADMIN_PROHIB. The router MAY, however, randomly ignore the rejected messages. (See Section 9.6.)

## 7. Configurations

### 7.1. CCNinfo Reply Timeout

The [CCNinfo Reply Timeout] value is used to time out a CCNinfo Reply. The value for a router can be statically configured by the router's administrators/operators. The default value is 4 (seconds). The [CCNinfo Reply Timeout] value SHOULD NOT be larger than 5 (seconds) and SHOULD NOT be lower than 2 (seconds).

### 7.2. HopLimit in Fixed Header

If a CCNinfo user does not specify the HopLimit value in a fixed header for a Request message as the HopLimit, the HopLimit is set to 32. Note that 0 HopLimit is an invalid Request; hence the router in this case follows the way defined in Section 6.3.

### 7.3. Access Control

A router MAY configure the valid or invalid networks to enable an access control. The access control can be defined per name prefix, such as "who can retrieve which name prefix". See Section 9.2.





## 8. Diagnosis and Analysis

### 8.1. Number of Hops

A CCNinfo Request message is forwarded in a hop-by-hop manner and each forwarding router appended its own Report block. We can then verify the number of hops to reach the content forwarder or the publisher.

### 8.2. Caching Router Identification

It is possible to identify the routers in the path from the CCNinfo user to the content forwarder, while some routers may hide their identifier (e.g., IP address) with all-zeros in the Report blocks (Section 9.1).

### 8.3. TTL or Hop Limit

By taking the HopLimit from the content forwarder and forwarding TTL threshold over all hops, it is possible to discover the TTL or hop limit required for the content forwarder to reach the CCNinfo user.

### 8.4. Time Delay

If the routers have synchronized clocks, it is possible to estimate propagation and queuing delay from the differences between the timestamps at successive hops. However, this delay includes control processing overhead, so is not necessarily indicative of the delay

that data traffic would experience.

### 8.5. Path Stretch

By getting the path stretch "d / P", where "d" is the hop count of the data and "P" is the hop count from the consumer to the publisher, we can measure the improvement in path stretch in various cases, such as different caching and routing algorithms. We can then facilitate investigation of the performance of the protocol.

### 8.6. Cache Hit Probability

CCNinfo can show the number of received interests per cache or chunk on a router. By this, CCNinfo measures the content popularity (i.e., the number of accesses for each content/cache), and you can investigate the routing/caching strategy in networks.





## 9. Security Considerations

This section addresses some of the security considerations.

### 9.1. Policy-Based Information Provisioning for Request

Although CCNinfo gives excellent troubleshooting cues, some network administrators or operators may not want to disclose everything about their network to the public, or may wish to securely transmit private information to specific members of their networks. CCNinfo provides policy-based information provisioning allowing network administrators to specify their response policy for each router.

The access policy regarding "who is allowed to retrieve" and/or "what kind of information" can be defined for each router. For the former access policy, routers having the specified content MAY examine the signature enclosed in the Request message and decide whether they should notify the content information in the Reply or not. If the routers decide to not notify the content information, they MUST reply the CCNinfo Reply with the ReturnCode of ADMIN_PROHIB without appending any Reply (sub-)block TLV. For the latter policy, the

permission, whether (1) All (all cache information is disclosed), (2) Partial (cache information with the particular name prefix can (or cannot) be disclosed), or (3) Deny (no cache information is disclosed), is defined at routers.

On the other hand, we entail that each router does not disrupt forwarding CCNinfo Request and Reply messages. When a Request message is received, the router SHOULD insert Report block if the ReturnCode is NO_ERROR. Here, according to the policy configuration, the Node Identifier field in the Report block MAY be null (i.e., all-zeros), but the Request Arrival Time field SHOULD NOT be null. At last, the router SHOULD forward the Request message to the upstream router toward the content forwarder if the ReturnCode is kept with NO_ERROR.

## 9.2. Filtering of CCNinfo Users Located in Invalid Networks

A router MAY support an access control mechanism to filter out Requests from invalid CCNinfo users. For it, invalid networks (or domains) could, for example, be configured via a list of allowed/ disallowed networks (as seen in Section 7.3). If a Request is received from the disallowed network (according to the Node Identifier in the Request block), the Request MUST NOT be processed and the Reply with the ReturnCode of INFO_HIDDEN may be used to note that. The router MAY, however, perform rate-limited logging of such events.





## 9.3. Topology Discovery

CCNinfo can be used to discover actively-used topologies. If a network topology is a secret, CCNinfo Requests SHOULD be restricted at the border of the domain, using the ADMIN_PROHIB return code.

## 9.4. Characteristics of Content

CCNinfo can be used to discover what publishers are sending to what kinds of contents. If this information is a secret, CCNinfo Requests SHOULD be restricted at the border of the domain, using the ADMIN_PROHIB return code.

## 9.5. Longer or Shorter CCNinfo Reply Timeout

Routers can configure the CCNinfo Reply Timeout (Section 7.1), which is the allowable timeout value to keep the PIT entry. If routers configure the longer timeout value, there may be an attractive attack vector against PIT memory. Moreover, especially when the full discovery request option (Section 5.3) is specified for the CCNinfo Request, a number of Reply messages may come back and cause a response storm. (See Section 9.7 for rate limiting to avoid the storm). In order to avoid DoS attacks, routers may configure the timeout value, which is shorter than the user-configured CCNinfo timeout value. However, if it is too short, the Request may be timed out and the CCNinfo user does not receive the all Replies and only retrieves the partial path information (i.e., information about part of the tree).

There may be the way to allow for incremental exploration (i.e., to explore the part of the tree the previous operation did not explore), whereas discussing such mechanism is out of scope of this document.

## 9.6. Limiting Request Rates

A router may limit CCNinfo Requests by ignoring some of the consecutive messages. The router MAY randomly ignore the received messages to minimize the processing overhead, i.e., to keep fairness in processing requests, or prevent traffic amplification. No error is returned. The rate limit is left to the router's implementation.

## 9.7. Limiting Reply Rates

CCNinfo supporting multipath forwarding may result in one Request returning multiple Reply messages. In order to prevent abuse, the routers in the traced path MAY need to rate-limit the Replies. No error is returned. The rate limit function is left to the router's implementation.





## 9.8. Adjacency Verification

It is assumed that CCNinfo Request and Reply messages are forwarded by adjacent neighbor nodes or routers. Defining the secure way to verify the adjacency cannot rely on the way specified in CCNx message format or semantics, yet specifying the mechanism to validate adjacent neighbor routers is out of scope of this document. An

adjacency verification mechanism and the corresponding TLV for adjacency verification using hop-by-hop TLV header such as [8] is the potential way and will be defined in a separate document.

## 10. Acknowledgements

The authors would like to thank Spyridon Mastorakis, Ilya Moiseenko, David Oran, and Thierry Turletti for their valuable comments and suggestions on this document.

## 11. References

### 11.1. Normative References

[1] Mosko, M., Solis, I., and C. Wood, "CCNx Messages in TLV Format", draft-irtf-icnrg-ccnxmessages-09 (work in progress), January 2019.

[2] Mosko, M., Solis, I., and C. Wood, "CCNx Semantics", draft-irtf-icnrg-ccnxsemantics-10 (work in progress), January 2019.

[3] Bradner, S., "Key words for use in RFCs to indicate requirement levels", RFC 2119, March 1997.

[4] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 8200, July 2017.

### 11.2. Informative References

[5] Asaeda, H., Matsuzono, K., and T. Turletti, "Contrace: A Tool for Measuring and Tracing Content-Centric Networks", IEEE Communications Magazine, Vol.53, No.3, pp.182-188, March 2015.

[6] Malkin, G., "Traceroute Using an IP Option", RFC 1393, January 1993.

[7] Asaeda, H., Mayer, K., and W. Lee, "Mtrace Version 2: Traceroute Facility for IP Multicast", RFC 8487, October 2018.

[8] Li, R., Asaeda, H., and J. Wu, "DCAuth: Data-Centric Authentication for Secure In-Network Big-Data Retrieval", IEEE Transactions on Network Science and Engineering (TNSE) , October 2018.

## Appendix A. ccninfo Command and Options

The ccninfo command enables the CCNinfo user to investigate the routing path based on the name prefix of the content (e.g., ccn:/news/today). The name prefix is mandatory but exclusive options; that is, only one of them should be used with the ccninfo command at once.

The usage of ccninfo command is as follows:

```
Usage: ccninfo [-f] [-n] [-o] [-r hop_count] [-s hop_count]
       name_prefix
```

name_prefix
: Prefix name of content (e.g., ccn:/news/today) or exact name of content (e.g., ccn:/news/today/Chunk=10) the CCNinfo user wants to trace.

f option
: This option enables "full discovery request"; routers ignore the forwarding strategy and send CCNinfo Requests to multiple upstream routers simultaneously. The CCNinfo user could then trace the all potential forwarding paths.

n option
: This option can be specified if a CCNinfo user only needs the routing path information to the specified content/cache and RTT between CCNinfo user and content forwarder; therefore, cache information is not given.

o option
: This option enables to trace the path to the content publisher. Each router along the path to the publisher inserts each Report block and forwards the Request message. It does not send Reply even if it caches the specified content. FHR that attaches the publisher (who has the complete set of content and is not a caching router) replies the Reply message.

r option
: Number of traced routers. If the CCNinfo user specifies this option, only the specified number of hops from the CCNinfo user trace the Request; each router inserts its own Report block and forwards the Request message to the upstream router(s), and the

last router stops the trace and sends the Reply message back to the CCNinfo user. This value is set in the "HopLimit" field located in the fixed header of the Request. For example, when the CCNinfo user invokes the CCNinfo command with this option such as "-r 3", only three routers along the path examine their path and cache information. If there is a caching router or FHR within the hop count along the path, the caching router or FHR sends back the Reply message and terminates the trace request. If the last router does not have the corresponding cache, it replies the Reply message with NO_INFO return code (described in Section 3.1) with no Reply block TLV inserted. The Request messages are terminated at FHR; therefore, although the maximum value for this option a CCNinfo user can specify is 255, the Request messages should be in general reached at FHR within significantly lower than 255 hops.

s option
: Number of skipped routers. If the CCNinfo user specifies this option, the number of hops from the CCNinfo user simply forward the CCNinfo Request messages without adding its own Report block and without replying the Request, and the next upstream router starts the trace. This value is set in the "SkipHopCount" field located in the Request block TLV. For example, when the CCNinfo user invokes the CCNinfo command with this option such as "-s 3", the three upstream routers along the path only forwards the Request message, but does not append their Report blocks in the hop-by-hop headers and does not send the Reply messages even though they have the corresponding cache. The Request messages are terminated at FHR; therefore, although the maximum value for this option a CCNinfo user can specify is 255, if the Request messages reaches FHR, the FHR silently discards the Request message and the request will be timed out.

## Authors' Addresses

Hitoshi Asaeda
National Institute of Information and Communications Technology
4-2-1 Nukui-Kitamachi
Koganei, Tokyo  184-8795
Japan

Email: asaeda@nict.go.jp

Atsushi Ooka
National Institute of Information and Communications Technology
4-2-1 Nukui-Kitamachi
Koganei, Tokyo  184-8795
Japan

Email: a-ooka@nict.go.jp

Xun Shao
Kitami Institute of Technology
165 Koen-cho
Kitami, Hokkaido  090-8507
Japan

Email: x-shao@mail.kitami-it.ac.jp