ICNRG
Internet-Draft
Intended status: Experimental
Expires: July 28, 2019

M. Mosko
PARC, Inc.
I. Solis
LinkedIn
C. Wood
University of California Irvine

January 24, 2019

CCNx Messages in TLV Format
draft-irtf-icnrg-ccnxmessages-09

Abstract

This document specifies the encoding of CCNx messages in a TLV packet format, including the TLV types used by each message element and the encoding of each value. The semantics of CCNx messages follow the encoding-independent CCNx Semantics specification.

This document is a product of the Information Centric Networking research group (ICNRG).

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 http://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 July 28, 2019.

Copyright Notice

Table of Contents

1. Introduction

This document specifies a Type-Length-Value (TLV) packet format and the TLV type and value encodings for CCNx messages. A full description of the CCNx network protocol, providing an encoding-free description of CCNx messages and message elements, may be found in [CCNSemantics]. CCNx is a network protocol that uses a hierarchical name to forward requests and to match responses to requests. It does not use endpoint addresses, such as Internet Protocol. Restrictions in a request can limit the response by the public key of the response's signer or the cryptographic hash of the response. Every CCNx forwarder along the path does the name matching and restriction checking. The CCNx protocol fits within the broader framework of Information Centric Networking (ICN) protocols [RFC7927].

This document describes a TLV scheme using a fixed 2-byte T and a fixed 2-byte L field. The rational for this choice is described in Section 5. Briefly, this choice avoids multiple encodings of the same value (aliases) and reduces the work of a validator to ensure compliance. Unlike some uses of TLV in networking, the each network hop must evaluate the encoding, so even small validation latencies at each hop could add up to a large overall forwarding delay. For very small packets or low throughput links, where the extra bytes may become a concern, one may use a TLV compression protocol, for example [compress] and [CCNxz].

This document specifies:

- The TLV packet format.
- The overall packet format for CCNx messages.
- The TLV types used by CCNx messages.
- The encoding of values for each type.
- Top level types that exist at the outermost containment.

- Interest TLVs that exist within Interest containment.

- Content Object TLVs that exist within Content Object containment.

This document is supplemented by this document:

- Message semantics: see [CCNSemantics] for the protocol operation regarding Interest and Content Object, including the Interest Return protocol.

- URI notation: see [CCNxURI] for the CCNx URI notation.

The type values in Section 4 represent the values in common usage today. These values may change pending IANA assignments. All type values are relative to their parent containers. For example, each level of a nested TLV structure might define a "type = 1" with a completely different meaning. In the following, we use the symbolic names defined in that section.

Packets are represented as 32-bit wide words using ASCII art. Due to the nested levels of TLV encoding and the presence of optional fields and variable sizes, there is no concise way to represent all possibilities. We use the convention that ASCII art fields enclosed by vertical bars "|" represent exact bit widths. Fields with a forward slash "/" are variable bit widths, which we typically pad out to word alignment for picture readability.

The document represents the consensus of the ICN RG. It is the first ICN protocol from the RG, created from the early CCNx protocol [nnc] with significant revision and input from the ICN community and RG members. The draft has received critical reading by several members of the ICN community and the RG. The authors and RG chairs approve of the contents. The document is sponsored under the IRTF and is not issued by the IETF and is not an IETF standard. This is an experimental protocol and may not be suitable for any specific application and the specification may change in the future.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Definitions

- Name: A hierarchically structured variable length identifier. It is an ordered list of path segments, which are variable length octet strings. In human-readable form, it is represented in URI

format as ccnx:/path/part. There is no host or query string. See [CCNxURI] for complete details.

- Interest: A message requesting a Content Object with a matching Name and other optional selectors to choose from multiple objects with the same Name. Any Content Object with a Name and attributes that matches the Name and optional selectors of the Interest is said to satisfy the Interest.

- Content Object: A data object sent in response to an Interest request. It has an optional Name and a content payload that are bound together via cryptographic means.

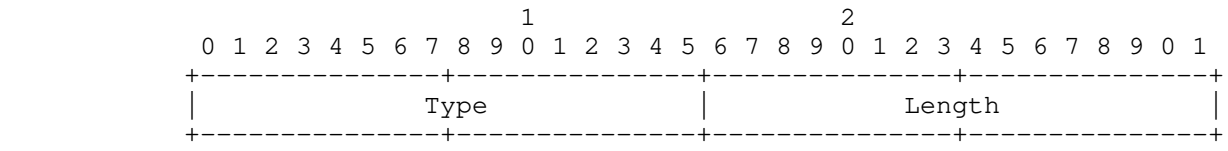
3. Type-Length-Value (TLV) Packets

We use 16-bit Type and 16-bit Length fields to encode TLV based packets. This provides 64K different possible types and value field lengths of up to 64KiB. With 64K possible types at each level of TLV encoding, there should be sufficient space for basic protocol types, while also allowing ample room for experimentation, application use, vendor extensions, and growth. This encoding does not allow for jumbo packets beyond 64 KiB total length. If used on a media that allows for jumbo frames, we suggest defining a media adapation envelope that allows for multiple smaller frames.

There are several global TLV definitions that we reserve at all hierarchical contexts. The TLV types in the range 0x1000 - 0x1FFF are reserved for experimental use. The TLV type T_ORG is also reserved for vendor extensions (see Section 3.3.2). The TLV type T_PAD is used to optionally pad a field out to some desired alignment.

Abbrev	Name	Description
T_ORG	Vendor Specific Information (Section 3.3.2)	Information specific to a vendor implementation (see below).
T_PAD	Padding (Section 3.3.1)	Adds padding to a field (see below).
n/a	Experimental	Experimental use.

Table 1: Reserved TLV Types



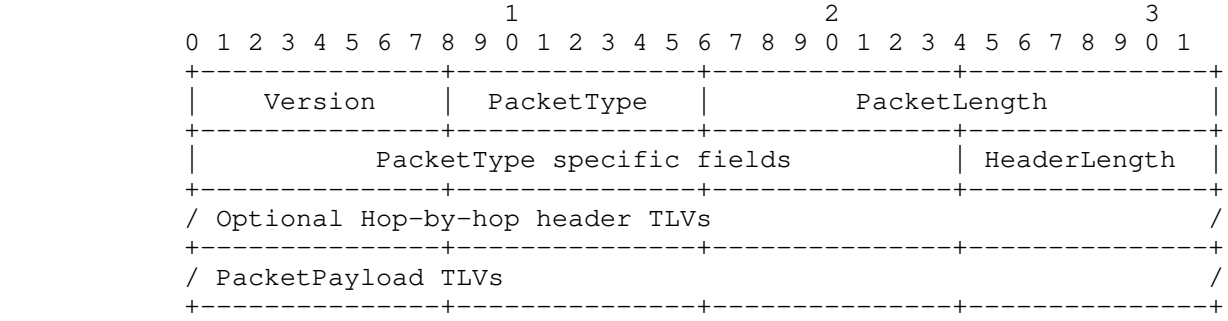
The Length field contains the length of the Value field in octets. It does not include the length of the Type and Length fields. The length MAY be zero.

TLV structures are nestable, allowing the Value field of one TLV structure to contain additional TLV structures. The enclosing TLV structure is called the container of the enclosed TLV.

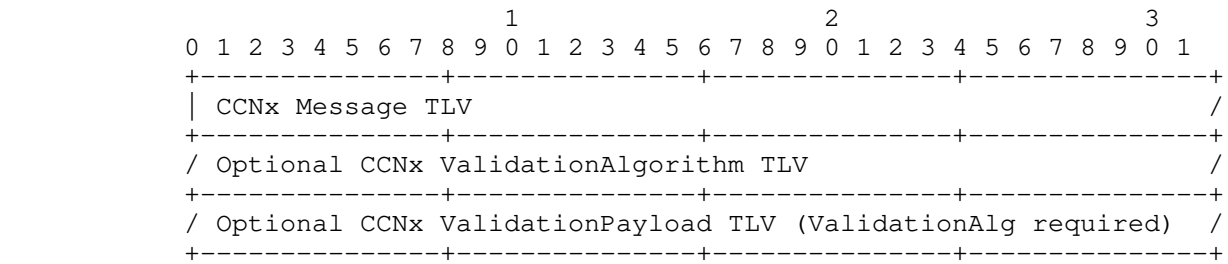
Type values are context-dependent. Within a TLV container, one may re-use previous type values for new context-dependent purposes.

3.1. Overall packet format

Each packet includes the 8 byte fixed header, described below, followed by a set of TLV fields. These fields are optional hop-by-hop headers and the Packet Payload.



The packet payload is a TLV encoding of the CCNx message, followed by optional Validation TLVs.



This document describes the Version "1" TLV encoding.

After discarding the fixed and hop-by-hop headers the remaining PacketPayload should be a valid protocol message. Therefore, the PacketPayload always begins with 4 bytes of type-length that specifies the protocol message (whether it is an Interest, Content Object, or other message type) and its total length. The embedding of a self-sufficient protocol data unit inside the fixed and hop-by-hop headers allows a network stack to discard the headers and operate only on the embedded message. It also de-couples the PacketType field -- which specifies how to forward the packet -- from the PacketPayload.

The range of bytes protected by the Validation includes the CCNx Message and the ValidationAlgorithm.

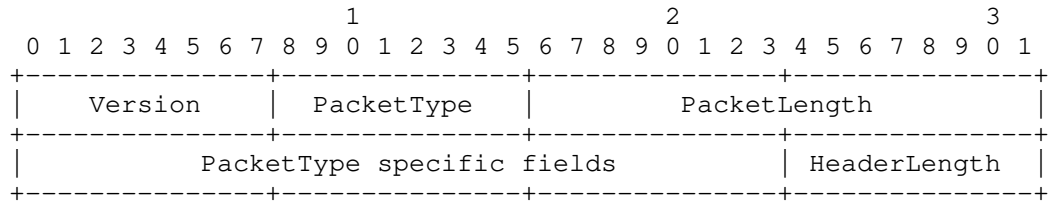
The ContentObjectHash begins with the CCNx Message and ends at the tail of the packet.

3.2. Fixed Headers

CCNx messages begin with an 8 byte fixed header (non-TLV format). The HeaderLength field represents the combined length of the Fixed and Hop-by-hop headers. The PacketLength field represents the entire Packet length from the first byte of Version to the last byte of the packet.

A specific PacketType may assign meaning to the "PacketType specific fields," which are otherwise reserved. For the three defined PacketTypes (Interest, ContentObject, and InterestReturn), we define those values in this document.

The PacketPayload of a CCNx packet is the protocol message itself. The Content Object Hash is computed over the PacketPayload only, excluding the fixed and hop-by-hop headers as those might change from hop to hop. Signed information or Similarity Hashes should not include any of the fixed or hop-by-hop headers. The PacketPayload should be self-sufficient in the event that the fixed and hop-by-hop headers are removed.



- Version: defines the version of the packet.
- HeaderLength: The length of the fixed header (8 bytes) and hop-by-hop headers. The minimum value MUST be "8".
- PacketType: describes forwarder actions to take on the packet.
- PacketLength: Total octets of packet including all headers (fixed header plus hop-by-hop headers) and protocol message.
- PacketType Specific Fields: specific PacketTypes define the use of these bits.

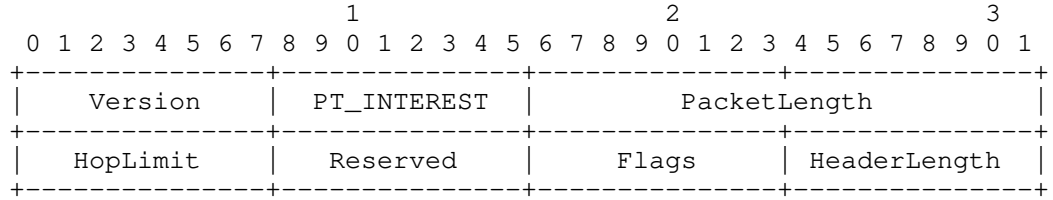
The PacketType field indicates how the forwarder should process the packet. A Request Packet (Interest) has PacketType PT_INTEREST, a Response (Content Object) has PacketType PT_CONTENT, and an InterestReturn has PacketType PT_RETURN.

HeaderLength is the number of octets from the start of the packet (Version) to the end of the hop-by-hop headers. PacketLength is the number of octets from the start of the packet to the end of the packet. Both lengths have a minimum value of 8 (the fixed header itself).

The PacketType specific fields are reserved bits whose use depends on the PacketType. They are used for network-level signaling.

3.2.1. Interest Fixed Header

If the PacketType is PT_INTEREST, it indicates that the PacketPayload should be processed as an Interest message. For this type of packet, the Fixed Header includes a field for a HopLimit as well as Reserved and Flags fields. The Reserved field MUST be set to 0 in an Interest - this field will be set to a return code in the case of an Interest Return. There are currently no Flags defined, so this field MUST be set to 0.



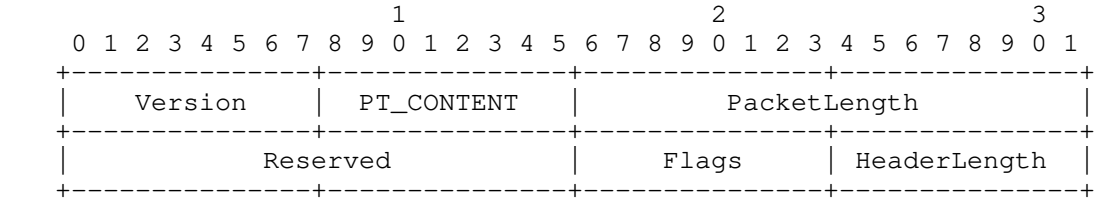
3.2.1.1. Interest HopLimit

For an Interest message, the HopLimit is a counter that is decremented with each hop. It limits the distance an Interest may travel on the network. The node originating the Interest MAY put in any value - up to the maximum of 255. Each node that receives an Interest with a HopLimit decrements the value upon reception. If the value is 0 after the decrement, the Interest MUST NOT be forwarded off the node.

It is an error to receive an Interest with a 0 hop-limit from a remote node.

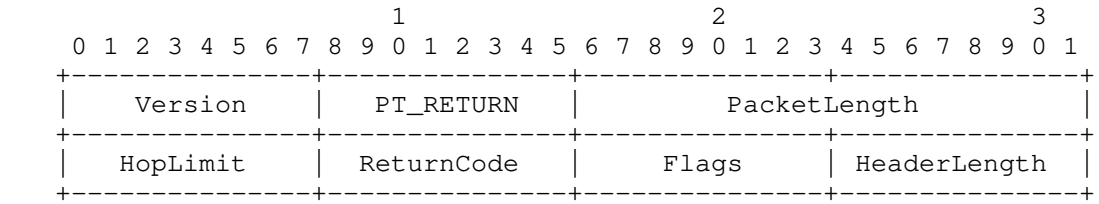
3.2.2. Content Object Fixed Header

If the PacketType is PT_CONTENT, it indicates that the PacketPayload should be processed as a Content Object message. A Content Object defines a Flags field, however there are currently no flags defined, so the Flags field must be set to 0.



3.2.3. InterestReturn Fixed Header

If the PacketType is PT_RETURN, it indicates that the PacketPayload should be processed as a returned Interest message. The only difference between this InterestReturn message and the original Interest is that the PacketType is changed to PT_RETURN and a ReturnCode is is put into the ReturnCode field. All other fields are unchanged from the Interest packet. The purpose of this encoding is to prevent packet length changes so no additional bytes are needed to return an Interest to the previous hop. See [CCNSemantics] for a protocol description of this packet type.



3.2.3.1. InterestReturn HopLimit

This is the original Interest's HopLimit, as received. It is the value before being decremented at the current node (i.e. the received value).

3.2.3.2. InterestReturn Flags

These are the original Flags as set in the Interest.

3.2.3.3. Return Code

The numeric value assigned to the return types is defined below. This value is set by the node creating the Interest Return.

A return code of "0" MUST NOT be used, as it indicates that the returning system did not modify the Return Code field.

Type	Return Type
T_RETURN_NO_ROUTE	No Route
T_RETURN_LIMIT_EXCEEDED	Hop Limit Exceeded
T_RETURN_NO_RESOURCES	No Resources
T_RETURN_PATH_ERROR	Path Error
T_RETURN_PROHIBITED	Prohibited
T_RETURN_CONGESTED	Congested
T_RETURN_MTU_TOO_LARGE	MTU too large
T_RETURN_UNSUPPORTED_HASH_RESTRICTION	Unsupported ContentObjectHashRestriction
T_RETURN_MALFORMED_INTEREST	Malformed Interest

Table 2: Return Codes

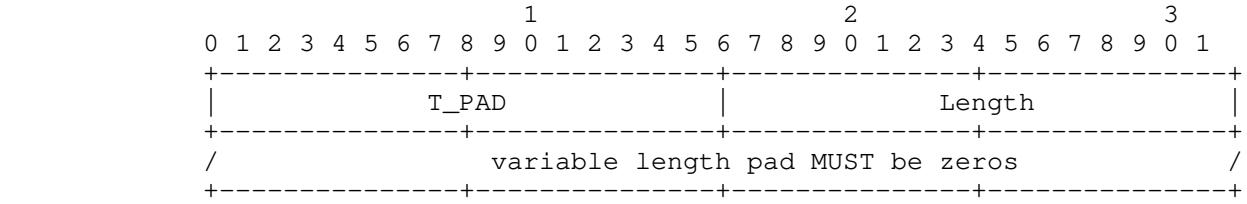
3.3. Global Formats

This section defines global formats that may be nested within other TLVs.

3.3.1. Pad

The pad type may be used by protocols that prefer word-aligned data. The size of the word may be defined by the protocol. Padding 4-byte words, for example, would use a 1-byte, 2-byte, and 3-byte Length. Padding 8-byte words would use a (0, 1, 2, 3, 5, 6, 7)-byte Length.

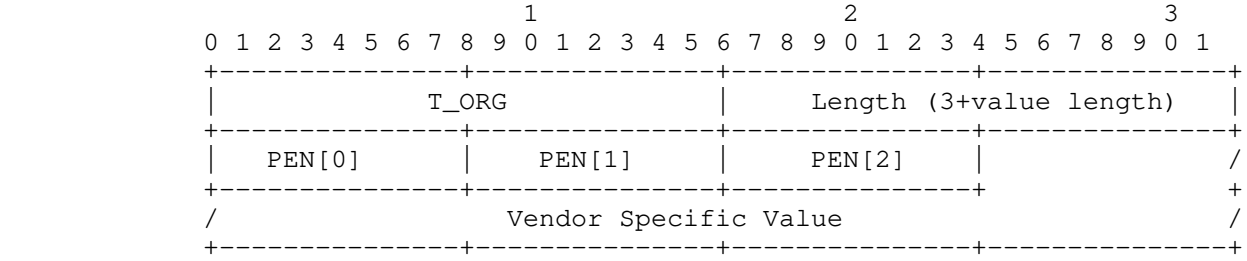
One MUST NOT pad inside a Name. Apart from that, a pad MAY be inserted after any other TLV in the CCNx Message or in the Validation Dependent Data In the remainder of this document, we will not show optional pad TLVs.



3.3.2. Organization Specific TLVs

Organization specific TLVs (also known as Vendor TLVs) MUST use the T_ORG type. The Length field is the length of the organization specific information plus 3. The Value begins with the 3 byte organization number derived from the last three digits of the IANA Private Enterprise Numbers [EpriseNumbers], followed by the organization specific information.

A T_ORG MAY be used as a path segment in a Name, in which case it is a regular path segment and is part of the regular name matching.



3.3.3. Hash Format

Hash values are used in several fields throughout a packet. This TLV encoding is commonly embedded inside those fields to specify the specific hash function used and it's value. Note that the reserved

TLV types are also reserved here for user-defined experimental functions.

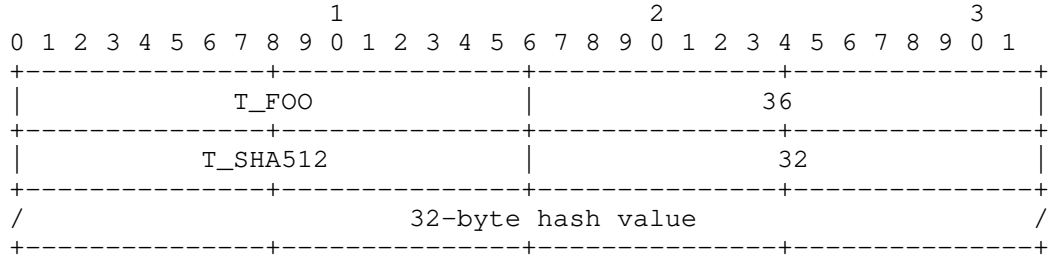
The LENGTH field of the hash value MUST be less than or equal to the hash function length. If the LENGTH is less than the full length, it is taken as the left LENGTH bytes of the hash function output. Only specified truncations are allowed, not arbitrary truncations.

This nested format is used because it allows binary comparison of hash values for certain fields without a router needing to understand a new hash function. For example, the KeyIdRestriction is bit-wise compared between an Interest's KeyIdRestriction field and a ContentObject's KeyId field. This format means the outer field values do not change with differing hash functions so a router can still identify those fields and do a binary comparison of the hash TLV without need to understand the specific hash used. An alternative approach, such as using T_KEYID_SHA512-256, would require each router keep an up-to-date parser and supporting user-defined hash functions here would explode the parsing state-space.

A CCNx entity MUST support the hash type T_SHA-256. An entity MAY support the remaining hash types.

Abbrev	Lengths (octets)
T_SHA-256	32
T_SHA-512	64, 32
n/a	Experimental TLV types

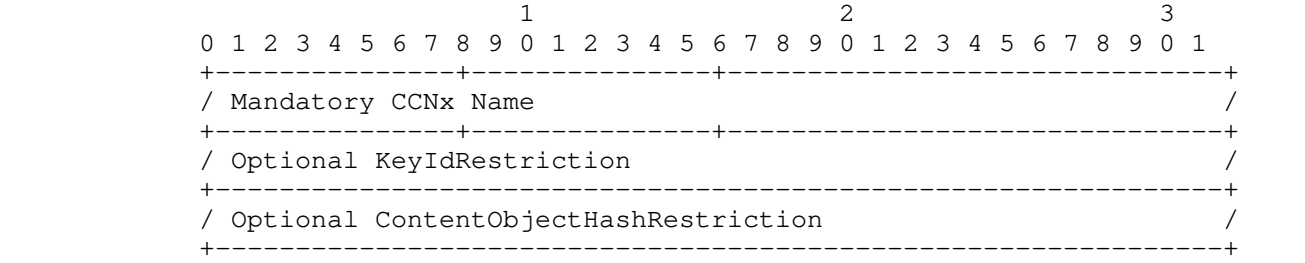
Table 3: CCNx Hash Functions



Example nesting inside type T_FOO

3.3.4. Link

A Link is the tuple: {Name, [KeyIdRestr], [ContentObjectHashRestr]}. It is a general encoding that is used in both the payload of a Content Object with PayloadType = "Link" and in the KeyLink field in a KeyLocator. A Link is essentially the body of an Interest.



3.4. Hop-by-hop TLV headers

Hop-by-hop TLV headers are unordered and meaning MUST NOT be attached to their ordering. Three hop-by-hop headers are described in this document:

Abbrev	Name	Description
T_INTLIFE	Interest Lifetime (Section 3.4.1)	The time an Interest should stay pending at an intermediate node.
T_CACHETIME	Recommended Cache Time (Section 3.4.2)	The Recommended Cache Time for Content Objects.
T_MSGHASH	Message Hash (Section 3.4.3)	The hash of the CCNx Message to end of packet using Section 3.3.3 format.

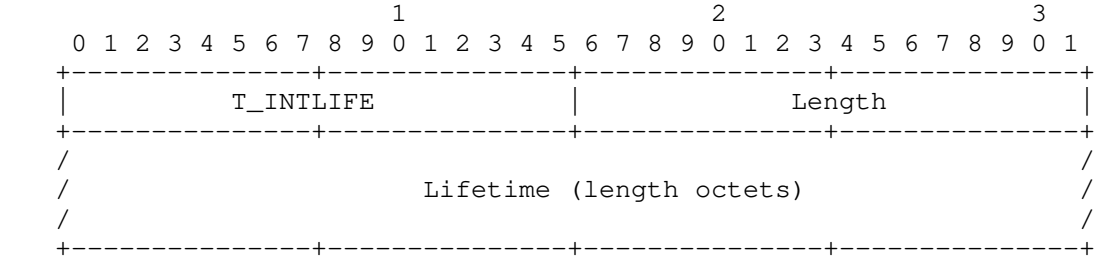
Table 4: Hop-by-hop Header Types

Additional hop-by-hop headers are defined in higher level specifications such as the fragmentation specification.

3.4.1. Interest Lifetime

The Interest Lifetime is the time that an Interest should stay pending at an intermediate node. It is expressed in milliseconds as an unsigned, network byte order integer.

A value of 0 (encoded as 1 byte %x00) indicates the Interest does not elicit a Content Object response. It should still be forwarded, but no reply is expected and a forwarder could skip creating a PIT entry.

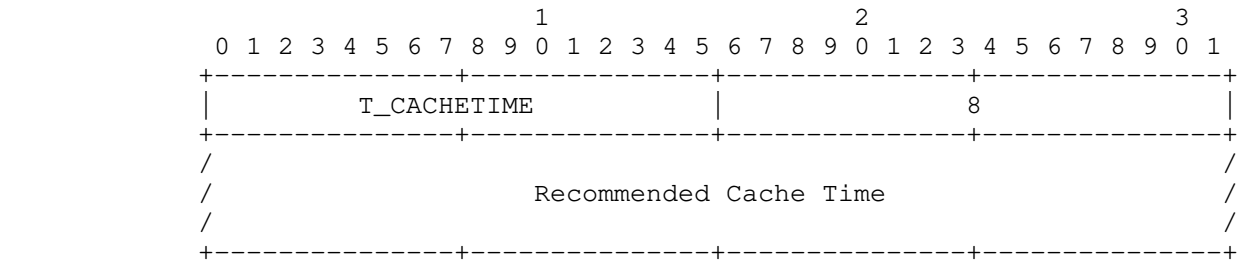


3.4.2. Recommended Cache Time

The Recommended Cache Time (RCT) is a measure of the useful lifetime of a Content Object as assigned by a content producer or upstream node. It serves as a guideline to the Content Store cache in determining how long to keep the Content Object. It is a recommendation only and may be ignored by the cache. This is in contrast to the ExpiryTime (described in Section 3.6.2.2.2) which takes precedence over the RCT and must be obeyed.

Because the Recommended Cache Time is an optional hop-by-hop header and not a part of the signed message, a content producer may re-issue a previously signed Content Object with an updated RCT without needing to re-sign the message. There is little ill effect from an attacker changing the RCT as the RCT serves as a guideline only.

The Recommended Cache Time (a millisecond timestamp) is a network byte ordered unsigned integer of the number of milliseconds since the epoch in UTC of when the payload expires. It is a 64-bit field.



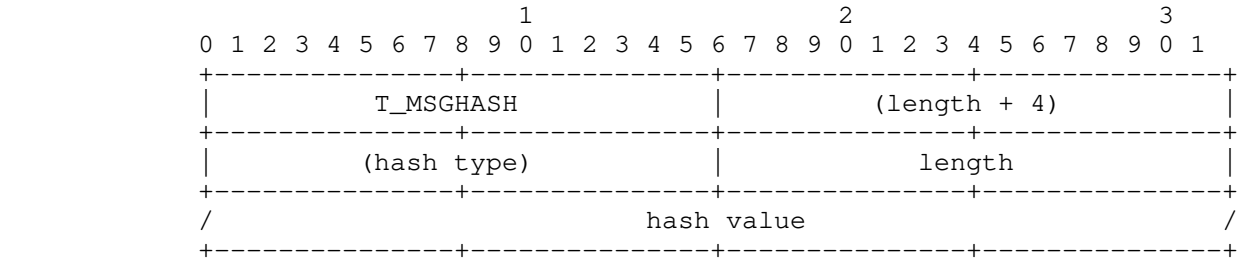
3.4.3. Message Hash

Within a trusted domain, an operator may calculate the message hash at a border device and insert that value into the hop-by-hop headers of a message. An egress device should remove the value. This permits intermediate devices within that trusted domain to match against a ContentObjectHashRestriction without calculating it at every hop.

The message hash is a cryptographic hash from the start of the CCNx Message to the end of the packet. It is used to match against the ContentObjectHashRestriction (Section 3.6.2.1.2). The Message Hash may be of longer length than an Interest's restriction, in which case the device should use the left bytes of the Message Hash to check against the Interest's value.

The Message Hash may only carry one hash type and there may only be one Message Hash header.

The Message Hash header is unprotected, so this header is only of practical use within a trusted domain, such as an operator's autonomous system.



Message Hash Header

3.5. Top-Level Types

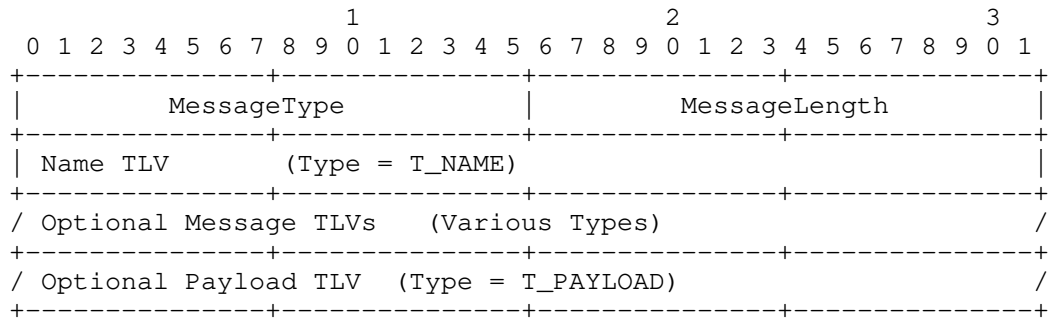
The top-level TLV types listed below exist at the outermost level of a CCNx protocol message.

Abbrev	Name	Description
T_INTEREST	Interest (Section 3.6)	An Interest MessageType.
T_OBJECT	Content Object (Section 3.6)	A Content Object MessageType
T_VALIDATION_ALG	Validation Algorithm (Section 3.6.4.1)	The method of message verification such as Message Integrity Check (MIC), a Message Authentication Code (MAC), or a cryptographic signature.
T_VALIDATION_PAYLOAD	Validation Payload (Section 3.6.4.2)	The validation output, such as the CRC32C code or the RSA signature.

Table 5: CCNx Top Level Types

3.6. CCNx Message

This is the format for the CCNx protocol message itself. The CCNx message is the portion of the packet between the hop-by-hop headers and the Validation TLVs. The figure below is an expansion of the "CCNx Message TLV" depicted in the beginning of Section 3. The CCNx message begins with MessageType and runs through the optional Payload. The same general format is used for both Interest and Content Object messages which are differentiated by the MessageType field. The first enclosed TLV of a CCNx Message is always the Name TLV. This is followed by an optional Message TLVs and an optional Payload TLV.



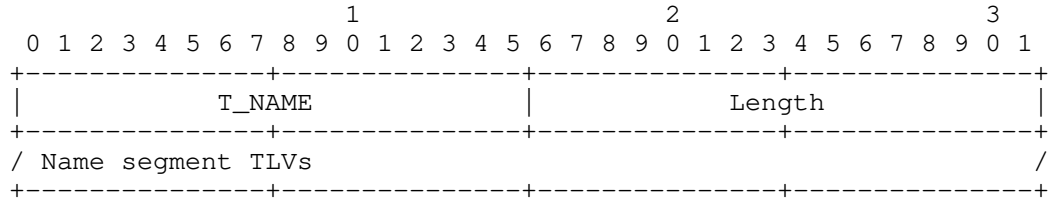
Abbrev	Name	Description
T_NAME	Name (Section 3.6.1)	The CCNx Name requested in an Interest or published in a Content Object.
T_PAYLOAD	Payload (Section 3.6.3)	The message payload.

Table 6: CCNx Message Types

3.6.1. Name

A Name is a TLV encoded sequence of segments. The table below lists the type values appropriate for these Name segments. A Name MUST NOT include PAD TLVs.

As described in CCNx Semantics [CCNSemantics], using the CCNx URI [CCNxURI] notation, a T_NAME with 0 length corresponds to ccnx:/ (the default route) and is distinct from a name with one zero length segment, such as ccnx:/NAME=. In the TLV encoding, ccnx:/ corresponds to T_NAME with 0 length, while ccnx:/NAME= corresponds to T_NAME with 4 length and T_NAMESEGMENT with 0 length.



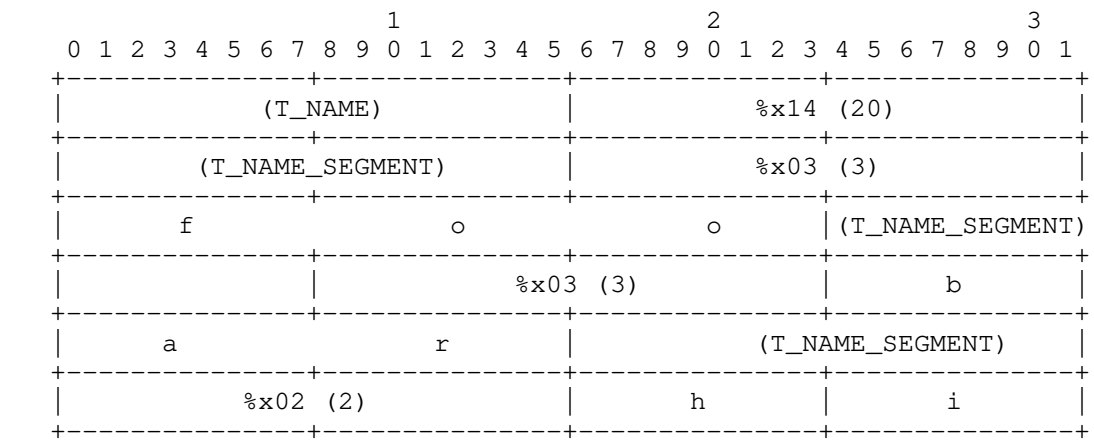
Symbolic Name	Name	Description
T_NAMESEGMENT	Name segment (Section 3.6.1.1)	A generic name Segment.
T_IPID	Interest Payload ID (Section 3.6.1.2)	An identifier that represents the Interest Payload field. As an example, the Payload ID might be a hash of the Interest Payload. This provides a way to differentiate between Interests based on their payloads without having to parse all the bytes of the payload itself; instead using only this Payload ID Name segment.
T_APP:00 - T_APP:4096	Application Components (Section 3.6.1.1)	Application-specific payload in a name segment. An application may apply its own semantics to the 4096 reserved types.

Table 7: CCNx Name Types

3.6.1.1. Name Segments

4096 special application payload name segments are allocated. These have application semantics applied to them. A good convention is to put the application's identity in the name prior to using these name segments.

For example, a name like "ccnx:/foo/bar/hi" would be encoded as:



3.6.1.2. Interest Payload ID

The InterestPayloadID is a name segment created by the origin of an Interest to represent the Interest Payload. This allows the proper multiplexing of Interests based on their name if they have different payloads. A common representation is to use a hash of the Interest Payload as the InterestPayloadID.

As part of the TLV 'value', the InterestPayloadID contains a one identifier of method used to create the InterestPayloadID followed by a variable length octet string. An implementation is not required to implement any of the methods to receive an Interest; the InterestPayloadID may be treated only as an opaque octet string for purposes of multiplexing Interests with different payloads. Only a device creating an InterestPayloadID name segment or a device verifying such a segment need to implement the algorithms.

It uses the Section 3.3.3 encoding of hash values.

In normal operations, we recommend displaying the InterestPayloadID as an opaque octet string in a CCNx URI, as this is the common denominator for implementation parsing.

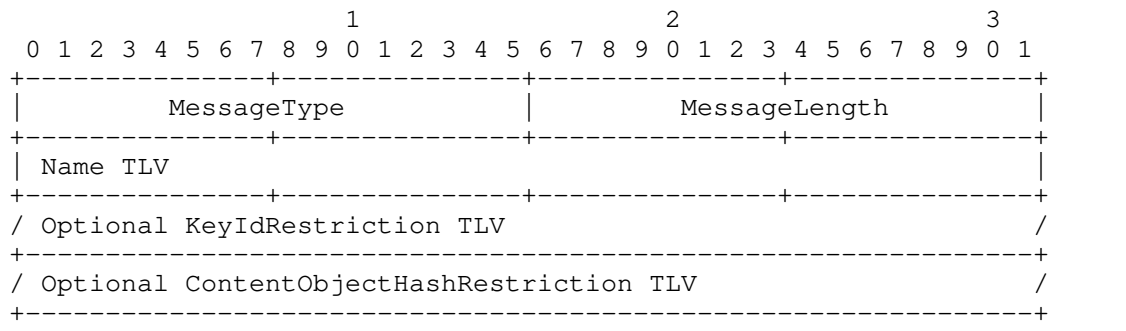
The InterestPayloadID, even if it is a hash, should not convey any security context. If a system requires confirmation that a specific entity created the InterestPayload, it should use a cryptographic signature on the Interest via the ValidationAlgorithm and ValidationPayload or use its own methods inside the Interest Payload.

3.6.2. Message TLVs

Each message type (Interest or Content Object) is associated with a set of optional Message TLVs. Additional specification documents may extend the types associated with each.

3.6.2.1. Interest Message TLVs

There are two Message TLVs currently associated with an Interest message: the KeyIdRestriction selector and the ContentObjectHashRestr selector are used to narrow the universe of acceptable Content Objects that would satisfy the Interest.



Abbrev	Name	Description
T_KEYIDRESTR	KeyIdRestriction (Section 3.6.2.1.1)	A Section 3.3.3 representation of the KeyId
T_OBJHASHRESTR	ContentObjectHashRestriction (Section 3.6.2.1.2)	A Section 3.3.3 representation of the hash of the specific Content Object that would satisfy the Interest.

Table 8: CCNx Interest Message TLV Types

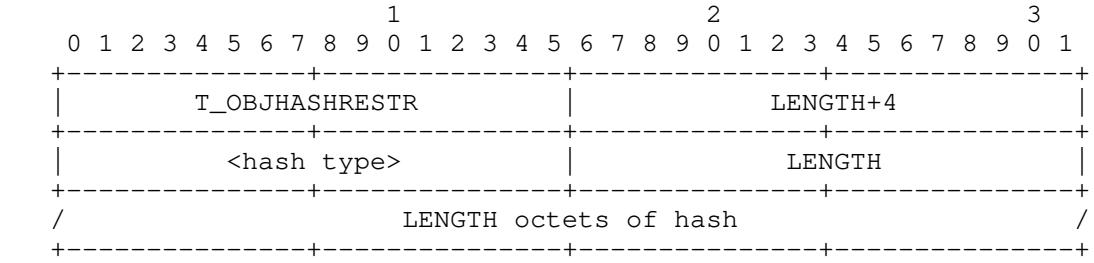
3.6.2.1.1. KeyIdRestriction

An Interest MAY include a KeyIdRestriction selector. This ensures that only Content Objects with matching KeyIds will satisfy the Interest. See Section 3.6.4.1.4.1 for the format of a KeyId.

3.6.2.1.2. ContentObjectHashRestriction

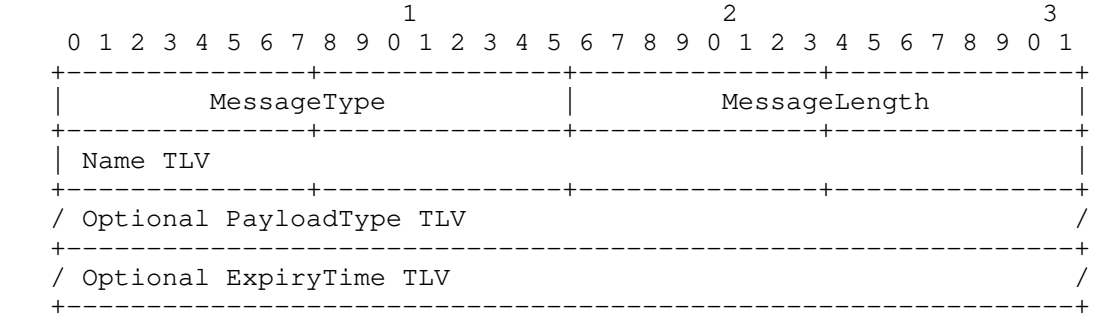
An Interest MAY contain a ContentObjectHashRestriction selector. This is the hash of the Content Object - the self-certifying name restriction that must be verified in the network, if an Interest carried this restriction. It is calculated from the beginning of the CCNx Message to the end of the packet. The LENGTH MUST be from one of the allowed values for that hash (see Section 3.3.3).

The ContentObjectHashRestriction SHOULD be of type T_SHA-256 and of length 32 bytes.



3.6.2.2. Content Object Message TLVs

The following message TLVs are currently defined for Content Objects: PayloadType (optional) and ExpiryTime (optional).



Abbrev	Name	Description
T_PAYLDTYPE	PayloadType (Section 3.6.2.2.1)	Indicates the type of Payload contents.
T_EXPIRY	ExpiryTime (Section 3.6.2.2.2)	The time at which the Payload expires, as expressed in the number of milliseconds since the epoch in UTC. If missing, Content Object may be used as long as desired.

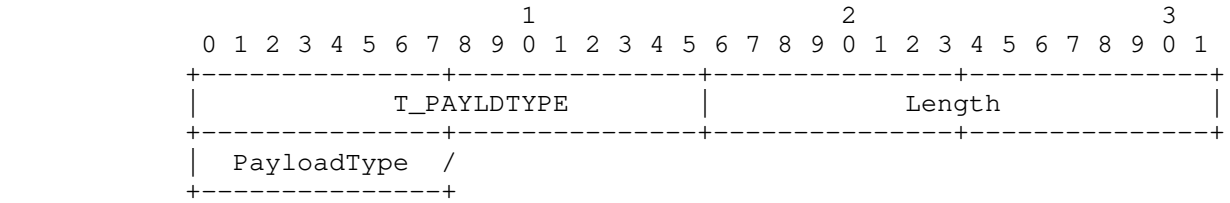
Table 9: CCNx Content Object Message TLV Types

3.6.2.2.1. PayloadType

The PayloadType is a network byte order integer representing the general type of the Payload TLV.

- T_PAYLOADTYPE_DATA: Data (possibly encrypted)
- T_PAYLOADTYPE_KEY: Key
- T_PAYLOADTYPE_LINK: Link

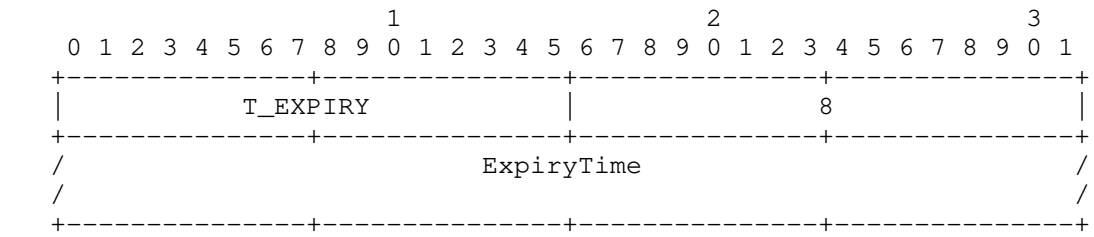
The Data type indicate that the Payload of the ContentObject is opaque application bytes. The Key type indicates that the Payload is a DER encoded public key. The Link type indicates that the Payload is one or more Link (Section 3.3.4). If this field is missing, a "Data" type is assumed.



3.6.2.2.2. ExpiryTime

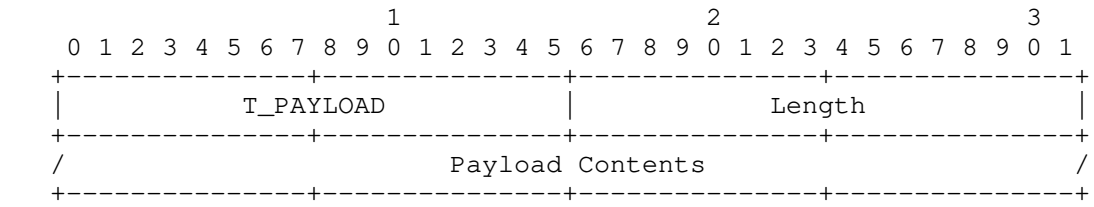
The ExpiryTime is the time at which the Payload expires, as expressed by a timestamp containing the number of milliseconds since the epoch in UTC. It is a network byte order unsigned integer in a 64-bit field. A cache or end system should not respond with a Content Object past its ExpiryTime. Routers forwarding a Content Object do

not need to check the ExpiryTime. If the ExpiryTime field is missing, the Content Object has no expressed expiration and a cache or end system may use the Content Object for as long as desired.



3.6.3. Payload

The Payload TLV contains the content of the packet. It MAY be of zero length. If a packet does not have any payload, this field MAY be omitted, rather than carrying a zero length.



3.6.4. Validation

Both Interests and Content Objects have the option to include information about how to validate the CCNx message. This information is contained in two TLVs: the ValidationAlgorithm TLV and the ValidationPayload TLV. The ValidationAlgorithm TLV specifies the mechanism to be used to verify the CCNx message. Examples include verification with a Message Integrity Check (MIC), a Message Authentication Code (MAC), or a cryptographic signature. The ValidationPayload TLV contains the validation output, such as the CRC32C code or the RSA signature.

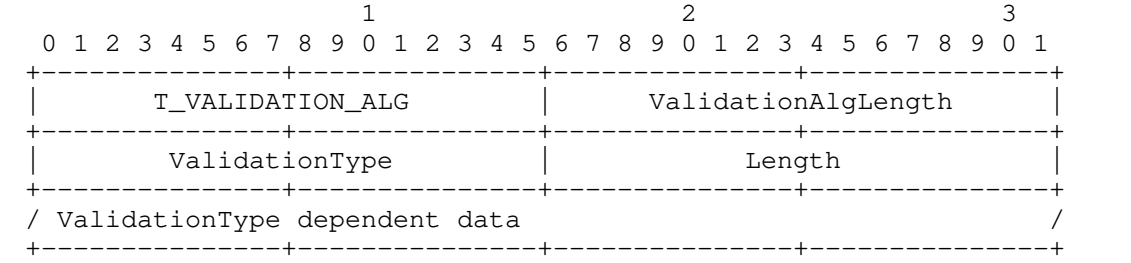
An Interest would most likely only use a MIC type of validation - a crc, checksum, or digest.

3.6.4.1. Validation Algorithm

The ValidationAlgorithm is a set of nested TLVs containing all of the information needed to verify the message. The outermost container has type = T_VALIDATION_ALG. The first nested TLV defines the specific type of validation to be performed on the message. The type

is identified with the "ValidationType" as shown in the figure below and elaborated in the table below. Nested within that container are the TLVs for any ValidationType dependent data, for example a Key Id, Key Locator etc.

Complete examples of several types may be found in Section 3.6.4.1.5



Abbrev	Name	Description
T_CRC32C	CRC32C (Section 3.6.4.1.1)	Castagnoli CRC32 (iSCSI, ext4, etc.), with normal form polynomial 0x1EDC6F41.
T_HMAC-SHA256	HMAC-SHA256 (Section 3.6.4.1.2)	HMAC (RFC 2104) using SHA256 hash.
T_RSA-SHA256	RSA-SHA256 (Section 3.6.4.1.3)	RSA public key signature using SHA256 digest.
EC-SECP-256K1	SECP-256K1 (Section 3.6.4.1.3)	Elliptic Curve signature with SECP-256K1 parameters (see [ECC]).
EC-SECP-384R1	SECP-384R1 (Section 3.6.4.1.3)	Elliptic Curve signature with SECP-384R1 parameters (see [ECC]).

Table 10: CCNx Validation Types

3.6.4.1.1. Message Integrity Checks

MICs do not require additional data in order to perform the verification. An example is CRC32C that has a "0" length value.

3.6.4.1.2. Message Authentication Checks

MACs are useful for communication between two trusting parties who have already shared private keys. Examples include an RSA signature of a SHA256 digest or others. They rely on a KeyId. Some MACs might use more than a KeyId, but those would be defined in the future.

3.6.4.1.3. Signature

Signature type Validators specify a digest mechanism and a signing algorithm to verify the message. Examples include RSA signature og a SHA256 digest, an Elliptic Curve signature with SECP-256K1 parameters, etc. These Validators require a KeyId and a mechanism for locating the publishers public key (a KeyLocator) - optionally a PublicKey or Certificate or KeyLink.

3.6.4.1.4. Validation Dependent Data

Different Validation Algorithms require access to different pieces of data contained in the ValidationAlgorithm TLV. As described above, Key Ids, Key Locators, Public Keys, Certificates, Links and Key Names all play a role in different Validation Algorithms. Any number of Validation Dependent Data containers can be present in a Validation Algorithm TLV.

Following is a table of CCNx ValidationType dependent data types:

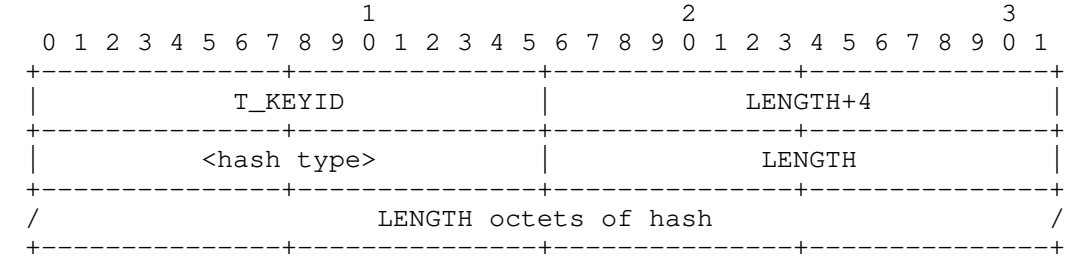
Abbrev	Name	Description
T_KEYID	SignerKeyId (Section 3.6.4.1.4.1)	An identifier of the shared secret or public key associated with a MAC or Signature.
T_PUBLICKEY	Public Key (Section 3.6.4.1.4.2)	DER encoded public key.
T_CERT	Certificate (Section 3.6.4.1.4.3)	DER encoded X509 certificate.
T_KEYLINK	KeyLink (Section 3.6.4.1.4.4)	A CCNx Link object.
T_SIGTIME	SignatureTime (Section 3.6.4.1.4.5)	A millsecond timestamp indicating the time when the signature was created.

Table 11: CCNx Validation Dependent Data Types

3.6.4.1.4.1. KeyId

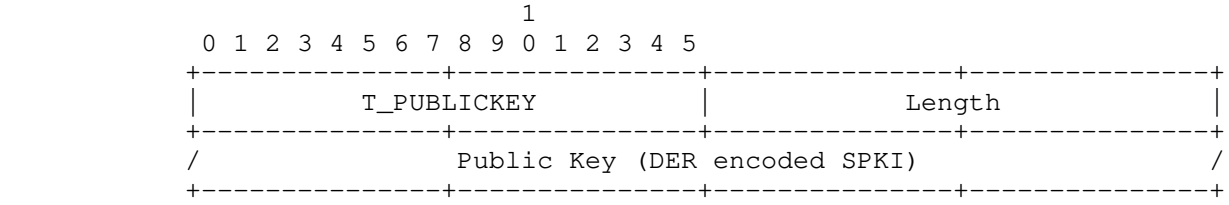
The KeyId is the publisher key identifier. It is similar to a Subject Key Identifier from X509 [RFC 5280, Section 4.2.1.2]. It should be derived from the key used to sign, such as from the SHA-256 hash of the key. It applies to both public/private key systems and to symmetric key systems.

The KeyId is represented using the Section 3.3.3. If a protocol uses a non-hash identifier, it should use one of the reserved values.

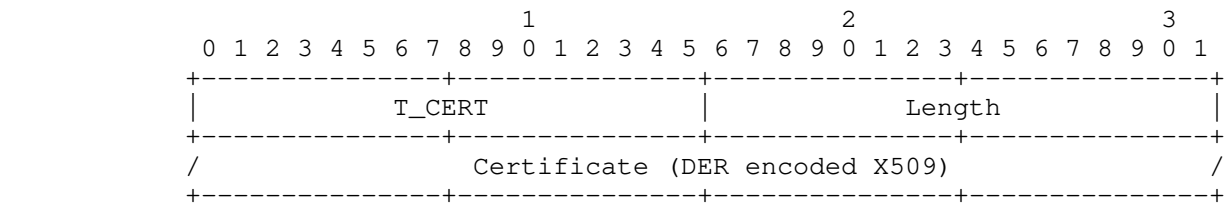


3.6.4.1.4.2. Public Key

A Public Key is a DER encoded Subject Public Key Info block, as in an X509 certificate.



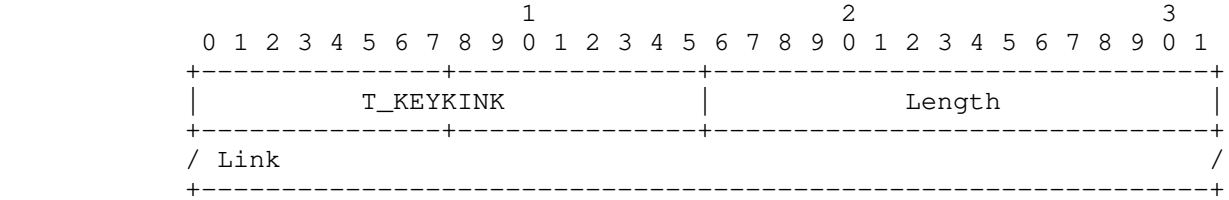
3.6.4.1.4.3. Certificate



3.6.4.1.4.4. KeyLink

A KeyLink type KeyLocator is a Link.

The KeyLink ContentObjectHashRestr, if included, is the digest of the Content Object identified by KeyLink, not the digest of the public key. Likewise, the KeyIdRestr of the KeyLink is the KeyId of the ContentObject, not necessarily of the wrapped key.

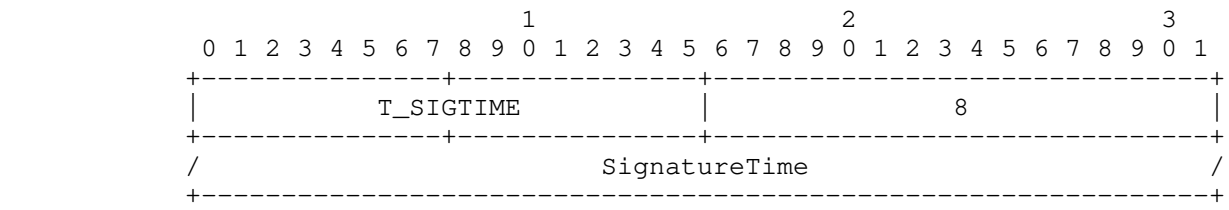


3.6.4.1.4.5. SignatureTime

The SignatureTime is a millisecond timestamp indicating the time at which a signature was created. The signer sets this field to the current time when creating a signature. A verifier may use this time to determine whether or not the signature was created during the validity period of a key, or if it occurred in a reasonable sequence with other associated signatures. The SignatureTime is unrelated to

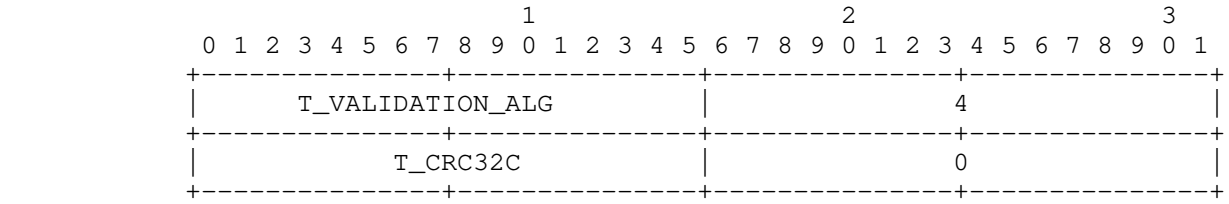
any time associated with the actual CCNx Message, which could have been created long before the signature. The default behavior is to always include a SignatureTime when creating an authenticated message (e.g. HMAC or RSA).

SignatureTime is a network byte ordered unsigned integer of the number of milliseconds since the epoch in UTC of when the signature was created. It is a fixed 64-bit field.

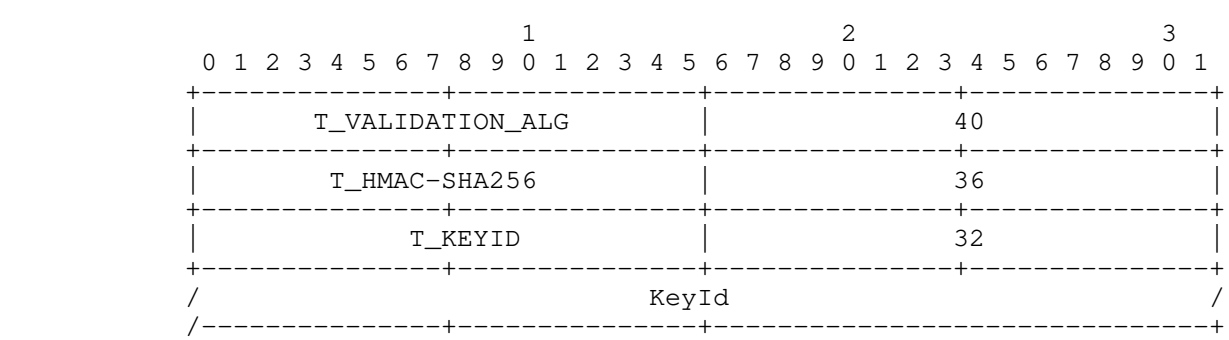


3.6.4.1.5. Validation Examples

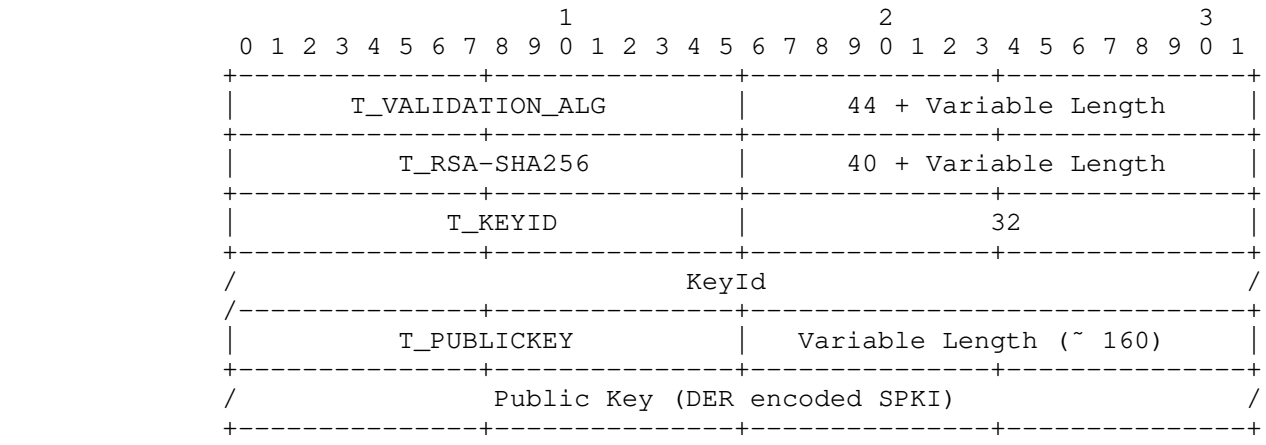
As an example of a MIC type validation, the encoding for CRC32C validation would be:



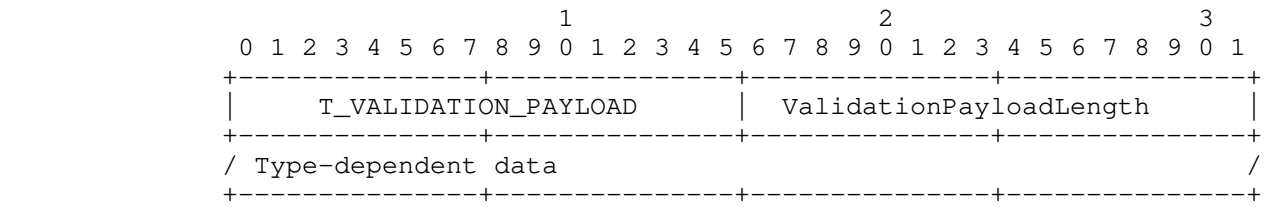
As an example of a MAC type validation, the encoding for an HMAC using a SHA256 hash would be:



As an example of a Signature type validation, the encoding for an RSA public key signing using a SHA256 digest and Public Key would be:



3.6.4.2. Validation Payload



The ValidationPayload contains the validation output, such as the CRC32C code or the RSA signature.

4. IANA Considerations

This section details each kind of protocol value that can be registered. Each type registry can be updated by incrementally expanding the type space, i.e., by allocating and reserving new types. As per [RFC5226] this section details the creation of the "CCNx Registry" and several sub-registries.

Property	Value
Name	CCNx Registry
Abbrev	CCNx

Registry Creation

4.1. Packet Type Registry

The following packet types should be allocated. A PacketType MUST be 1 byte. New packet types are allocated via "RFC Required" action.

Property	Value
Name	Packet Type Registry
Parent	CCNx Registry
Review process	RFC Required
Syntax	1 octet

Registry Creation

Type	Name	Reference
%x00	PT_INTEREST	Fixed Header Types (Section 3.2)
%x01	PT_CONTENT	Fixed Header Types (Section 3.2)
%x02	PT_RETURN	Fixed Header Types (Section 3.2)

Packet Type Namespace

4.2. Interest Return Code Registry

The following InterestReturn code types should be allocated.

Property	Value
Name	Interest Return Code
Parent	CCNx Registry
Review process	Specification Required
Syntax	1 octet

Registry Creation

Type	Name	Reference
%x00	Reserved	
%x01	T_RETURN_NO_ROUTE	Fixed Header Types (Section 3.2.3.3)
%x02	T_RETURN_LIMIT_EXCEEDED	Fixed Header Types (Section 3.2.3.3)
%x03	T_RETURN_NO_RESOURCES	Fixed Header Types (Section 3.2.3.3)
%x04	T_RETURN_PATH_ERROR	Fixed Header Types (Section 3.2.3.3)
%x05	T_RETURN_PROHIBITED	Fixed Header Types (Section 3.2.3.3)
%x06	T_RETURN_CONGESTED	Fixed Header Types (Section 3.2.3.3)
%x07	T_RETURN_MTU_TOO_LARGE	Fixed Header Types (Section 3.2.3.3)
%x08	T_RETURN_UNSUPPORTED_HASH_RESTRICTION	Fixed Header Types (Section 3.2.3.3)
%x09	T_RETURN_MALFORMED_INTEREST	Fixed Header Types (Section 3.2.3.3)

Interest Return Type Namespace

4.3. Hop-by-Hop Type Registry

The following hop-by-hop types should be allocated.

Property	Value
Name	Hop-by-Hop Type Registry
Parent	CCNx Registry
Review process	RFC Required
Syntax	2 octet TLV type

Registry Creation

Type	Name	Reference
%x0000	Reserved	
%x0001	T_INTLIFE	Hop-by-hop TLV headers (Section 3.4)
%x0002	T_CACHETIME	Hop-by-hop TLV headers (Section 3.4)
%x0003	T_MSGHASH	Hop-by-hop TLV headers (Section 3.4)
%x0004 - %x0007	Reserved	
%x0FFE	T_PAD	Pad (Section 3.3.1)
%x0FFF	T_ORG	Organization-Specific TLVs (Section 3.3.2)
%x1000-%x1FFF	Reserved	Experimental Use (Section 3)

Hop-by-Hop Type Namespace

4.4. Top-Level Type Registry

The following top-level types should be allocated.

Property	Value
Name	Top-Level Type Registry
Parent	CCNx Registry
Review process	RFC Required
Syntax	2 octet TLV type

Registry Creation

Type	Name	Reference
%x0000	Reserved	
%x0001	T_INTEREST	Top-Level Types (Section 3.5)
%x0002	T_OBJECT	Top-Level Types (Section 3.5)
%x0003	T_VALIDATION_ALG	Top-Level Types (Section 3.5)
%x0004	T_VALIDATION_PAYLOAD	Top-Level Types (Section 3.5)

Top-Level Type Namespace

4.5. Name Segment Type Registry

The following name segment types should be allocated.

Property	Value
Name	Name Segment Type Registry
Parent	CCNx Registry
Review process	Specification Required
Syntax	2 octet TLV type

Registry Creation

Type	Name	Reference
%x0000	Reserved	
%x0001	T_NAMESEGMENT	Name (Section 3.6.1)
%x0002	T_IPID	Name (Section 3.6.1)
%x0010 - %x0013	Reserved	Used in other drafts
%x0FFF	T_ORG	Organization-Specific TLVs (Section 3.3.2)
%x1000 - %x1FFF	T_APP:00 - T_APP:4096	Application Components (Section 3.6.1)

Name Segment Type Namespace

4.6. Message Type Registry

The following CCNx message segment types should be allocated.

Property	Value
Name	Message Type Registry
Parent	CCNx Registry
Review process	RFC Required
Syntax	2 octet TLV type

Registry Creation

Type	Name	Reference
%x0000	T_NAME	Message Types (Section 3.6)
%x0001	T_PAYLOAD	Message Types (Section 3.6)
%x0002	T_KEYIDRESTR	Message Types (Section 3.6)
%x0003	T_OBJHASHRESTR	Message Types (Section 3.6)
%x0005	T_PAYLDTYPE	Content Object Message Types (Section 3.6.2.2)
%x0006	T_EXPIRY	Content Object Message Types (Section 3.6.2.2)
%x0007 - %x000C	Reserved	Used in other RFC drafts
%x0FFE	T_PAD	Pad (Section 3.3.1)
%x0FFF	T_ORG	Organization-Specific TLVs (Section 3.3.2)
%x1000-%x1FFF	Reserved	Experimental Use (Section 3)

CCNx Message Type Namespace

4.7. Payload Type Registry

The following payload types should be allocated.

Property	Value
Name	PayloadType Registry
Parent	CCNx Registry
Review process	Specification Required
Syntax	Variable length unsigned integer

Registry Creation

Type	Name	Reference
%x00	T_PAYLOADTYPE_DATA	Payload Types (Section 3.6.2.2.1)
%x01	T_PAYLOADTYPE_KEY	Payload Types (Section 3.6.2.2.1)
%x02	T_PAYLOADTYPE_LINK	Payload Types (Section 3.6.2.2.1)

Payload Type Namespace

4.8. Validation Algorithm Type Registry

The following validation algorithm types should be allocated. Note: registration requires public specification of the algorithm.

Property	Value
Name	Validation Algorithm Type Registry
Parent	CCNx Registry
Review process	Specification Required
Syntax	2 octet TLV type

Registry Creation

Type	Name	Reference
%x0000	Reserved	
%x0001	Unassigned	
%x0002	T_CRC32C	Validation Algorithm (Section 3.6.4.1)
%x0003	Unassigned	
%x0004	T_HMAC-SHA256	Validation Algorithm (Section 3.6.4.1)
%x0005	T_RSA-SHA256	Validation Algorithm (Section 3.6.4.1)
%x0006	EC-SECP-256K1	Validation Algorithm (Section 3.6.4.1)
%x0007	EC-SECP-384R1	Validation Algorithm (Section 3.6.4.1)
%x0FFE	T_PAD	Pad (Section 3.3.1)
%x0FFF	T_ORG	Organization-Specific TLVs (Section 3.3.2)
%x1000-%x1FFF	Reserved	Experimental Use (Section 3)

Validation Algorithm Type Namespace

4.9. Validation Dependent Data Type Registry

The following validation dependent data types should be allocated.

Property	Value
Name	Validation Dependent Data Type Registry
Parent	CCNx Registry
Review process	RFC Required
Syntax	2 octet TLV type

Registry Creation

Type	Name	Reference
%x0000	Reserved	
%x0001 - %x0008	Unassigned	
%x0009	T_KEYID	Validation Dependent Data (Section 3.6.4.1.4)
%x000A	T_PUBLICKEYLOC	Validation Dependent Data (Section 3.6.4.1.4)
%x000B	T_PUBLICKEY	Validation Dependent Data (Section 3.6.4.1.4)
%x000C	T_CERT	Validation Dependent Data (Section 3.6.4.1.4)
%x000D	T_LINK	Validation Dependent Data (Section 3.6.4.1.4)
%x000E	T_KEYLINK	Validation Dependent Data (Section 3.6.4.1.4)
%x000F	T_SIGTIME	Validation Dependent Data (Section 3.6.4.1.4)
%x0FFF	T_ORG	Organization-Specific TLVs (Section 3.3.2)
%x1000-%x1FFF	Reserved	Experimental Use (Section 3)

Validation Dependent Data Type Namespace

4.10. Hash Function Type Registry

The following CCNx hash function types should be allocated. Note: registration requires public specification of the algorithm.

Property	Value
Name	Hash Function Type Registry
Parent	CCNx Registry
Review process	Specification Required
Syntax	2 octet TLV type

Registry Creation

Type	Name	Reference
%x0000	Reserved	
%x0001	T_SHA-256	Hash Format (Section 3.3.3)
%x0002	T_SHA-512	Hash Format (Section 3.3.3)
%x0FFF	T_ORG	Organization-Specific TLVs (Section 3.3.2)
%x1000-%x1FFF	Reserved	Experimental Use (Section 3)

CCNx Hash Function Type Namespace

5. Security Considerations

The CCNx protocol is a layer 3 network protocol, which may also operate as an overlay using other transports, such as UDP or other tunnels. It includes intrinsic support for message authentication via a signature (e.g. RSA or elliptic curve) or message authentication code (e.g. HMAC). In lieu of an authenticator, it may instead use a message integrity check (e.g. SHA or CRC). CCNx does not specify an encryption envelope, that function is left to a high-layer protocol (e.g. [esic]).

The CCNx message format includes the ability to attach MICs (e.g. SHA-256 or CRC), MACs (e.g. HMAC), and Signatures (e.g. RSA or ECDSA) to all packet types. This does not mean that it is a good idea to use an arbitrary ValidationAlgorithm, nor to include computationally expensive algorithms in Interest packets, as that could lead to computational DoS attacks. Applications should use an

explicit protocol to guide their use of packet signatures. As a general guideline, an application might use a MIC on an Interest to detect unintentionally corrupted packets. If one wishes to secure an Interest, one should consider using an encrypted wrapper and a protocol that prevents replay attacks, especially if the Interest is being used as an actuator. Simply using an authentication code or signature does not make an Interests secure. There are several examples in the literature on how to secure ICN-style messaging [mobile] [ace].

As a layer 3 protocol, this document does not describe how one arrives at keys or how one trusts keys. The CCNx content object may include a public key embedded in the object or may use the PublicKeyLocator field to point to a public key (or public key certificate) that authenticates the message. One key exchange specification is CCNxKE [ccnxke] [mobile], which is similar to the TLS 1.3 key exchange except it is over the CCNx layer 3 messages. Trust is beyond the scope of a layer-3 protocol protocol and left to applications or application frameworks.

The combination of an ephemeral key exchange (e.g. CCNxKE [ccnxke]) and an encapsulating encryption (e.g. [esic]) provides the equivalent of a TLS tunnel. Intermediate nodes may forward the Interests and Content Objects, but have no visibility inside. It also completely hides the internal names in those used by the encryption layer. This type of tunneling encryption is useful for content that has little or no cache-ability as it can only be used by someone with the ephemeral key. Short term caching may help with lossy links or mobility, but long term caching is usually not of interest.

Broadcast encryption or proxy re-encryption may be useful for content with multiple uses over time or many consumers. There is currently no recommendation for this form of encryption.

The specific encoding of messages will have security implications. This document uses a type-length-value (TLV) encoding. We chose to compromise between extensibility and unambiguous encodings of types and lengths. Some TLVs use variable length T and variable length L fields to accomodate a wide gamut of values while trying to be byte-efficient. Our TLV encoding uses a fixed length 2-byte T and 2-byte L. Using a fixed-length T and L field solves two problems. The first is aliases. If one is able to encode the same value, such as 0x2 and 0x02, in different byte lengths then one must decide if they mean the same thing, if they are different, or if one is illegal. If they are different, then one must always compare on the buffers not the integer equivalents. If one is illegal, then one must validate the TLV encoding -- every field of every packet at every hop. If they are the same, then one has the second problem: how to specify

packet filters. For example, if a name has 6 name components, then there are 7 T's and 7 L's, each of which might have up to 4 representations of the same value. That would be 14 fields with 4 encodings each, or 1001 combinations. It also means that one cannot compare, for example, a name via a memory function as one needs to consider that any embedded T or L might have a different format.

The Interest Return message has no authenticator from the previous hop. Therefore, the payload of the Interest Return should only be used locally to match an Interest. A node should never forward that Interest payload as an Interest. It should also verify that it sent the Interest in the Interest Return to that node and not allow anyone to negate Interest messages.

Caching nodes must take caution when processing content objects. It is essential that the Content Store obey the rules outlined in [CCNSemantics] to avoid certain types of attacks. Unlike NDN, CCNx 1.0 has no mechanism to work around an undesired result from the network (there are no "excludes"), so if a cache becomes poisoned with bad content it might cause problems retrieving content. There are three types of access to content from a content store: unrestricted, signature restricted, and hash restricted. If an Interest has no restrictions, then the requester is not particular about what they get back, so any matching cached object is OK. In the hash restricted case, the requester is very specific about what they want and the content store (and every forward hop) can easily verify that the content matches the request. In the signature verified case (often used for initial manifest discovery), the requester only knows the KeyId that signed the content. It is this case that requires the closest attention in the content store to avoid amplifying bad data. The content store must only respond with a content object if it can verify the signature -- this means either the content object carries the public key inside it or the Interest carries the public key in addition to the KeyId. If that is not the case, then the content store should treat the Interest as a cache miss and let an endpoint respond.

A user-level cache could perform full signature verification by fetching a public key according to the PublicKeyLocator. That is not, however, a burden we wish to impose on the forwarder. A user-level cache could also rely on out-of-band attestation, such as the cache operator only inserting content that it knows has the correct signature.

The CCNx grammar allows for hash algorithm agility via the HashType. It specifies a short list of acceptable hash algorithms that should be implemented at each forwarder. Some hash values only apply to end systems, so updating the hash algorithm does not affect forwarders --

they would simply match the buffer that includes the type-length-hash buffer. Some fields, such as the ConObjHash, must be verified at each hop, so a forwarder (or related system) must know the hash algorithm and it could cause backward compatibility problems if the hash type is updated.

A CCNx name uses binary matching whereas a URI uses a case insensitive hostname. Some systems may also use case insensitive matching of the URI path to a resource. An implication of this is that human-entered CCNx names will likely have case or non-ASCII symbol mismatches unless one uses a consistent URI normalization to the CCNx name. It also means that an entity that registers a CCNx routable prefix, say ccnx:/example.com, would need separate registrations for simple variations like ccnx:/Example.com. Unless this is addressed in URI normalization and routing protocol conventions, there could be phishing attacks.

For a more general introduction to ICN-related security concerns and approaches, see [RFC7927] and [RFC7945]

6. References

6.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

6.2. Informative References

[ace] Shang, W., Yu, Y., Liang, T., Zhang, B., and L. Zhang, "NDN-ACE: Access control for constrained environments over named data networking", NDN Technical Report NDN-0036, 2015, <http://new.named-data.net/wp-content/uploads/2015/12/ndn-0036-1-ndn-ace.pdf>.

[CCNSemantics] Mosko, M., Solis, I., and C. Wood, "CCNx Semantics (Internet draft)", 2018, <https://www.ietf.org/id/draft-irtf-icnrg-ccnxsemantics-09.txt>.

[ccnxke] Mosko, M., Uzun, E., and C. Wood, "CCNx Key Exchange Protocol Version 1.0", 2017, <https://www.ietf.org/archive/id/draft-wood-icnrg-ccnxkeyexchange-02.txt>.

[CCNxURI] Mosko, M. and C. Wood, "The CCNx URI Scheme (Internet draft)", 2017, <http://tools.ietf.org/html/draft-mosko-icnrg-ccnxuri-02>.

[CCNxz] Mosko, M., "CCNxz TLV Header Compression Experimental Code", 2016-2018, <https://github.com/PARC/CCNxz>.

[compress] Mosko, M., "Header Compression for TLV-based Packets", 2016, <https://datatracker.ietf.org/meeting/interim-2016-icnrg-02/materials/slides-interim-2016-icnrg-2-7>.

[ECC] Certicom Research, "SEC 2: Recommended Elliptic Curve Domain Parameters", 2010, <http://www.secg.org/sec2-v2.pdf>.

[EpriseNumbers] IANA, "IANA Private Enterprise Numbers", 2015, <http://www.iana.org/assignments/enterprise-numbers/enterprise-numbers>.

[esic] Mosko, M. and C. Wood, "Encrypted Sessions In CCNx (ESIC)", 2017, <https://www.ietf.org/id/draft-wood-icnrg-esic-01.txt>.

[mobile] Mosko, M., Uzun, E., and C. Wood, "Mobile Sessions in Content-Centric Networks", IFIP Networking, 2017, <http://dl.ifip.org/db/conf/networking/networking2017/1570334964.pdf>.

[nnc] Jacobson, V., Smetters, D., Thornton, J., Plass, M., Briggs, N., and R. Braynard, "Networking Named Content", 2009, <http://dx.doi.org/10.1145/1658939.1658941>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <https://www.rfc-editor.org/info/rfc5226>.

[RFC7927] Kutscher, D., Eum, S., Pentikousis, K., Psaras, I., Corujo, D., Saucez, D., Schmidt, T., and M. Waehlisch, "Information-Centric Networking (ICN) Research Challenges", 2016, <https://trac.tools.ietf.org/html/rfc7927>.

[RFC7945] Pentikousis, K., Ohlman, B., Davies, E., Spirou, S., and G. Boggia, "Information-Centric Networking: Evaluation and Security Considerations", 2016, <https://trac.tools.ietf.org/html/rfc7945>.

Authors' Addresses

Marc Mosko
PARC, Inc.
Palo Alto, California 94304
USA

Phone: +01 650-812-4405
Email: marc.mosko@parc.com

Ignacio Solis
LinkedIn
Mountain View, California 94043
USA

Email: nsolis@linkedin.com

Christopher A. Wood
University of California Irvine
Irvine, California 92697
USA

Phone: +01 315-806-5939
Email: woodc1@uci.edu

ICNRG
Internet-Draft
Intended status: Experimental
Expires: July 28, 2019

M. Mosko
PARC, Inc.
I. Solis
LinkedIn
C. Wood
University of California Irvine

January 24, 2019

CCNx Semantics
draft-irtf-icnrg-ccnxsemantics-10

Abstract

This document describes the core concepts of the Content Centric Networking (CCNx) architecture and presents a network protocol based on two messages: Interests and Content Objects. It specifies the set of mandatory and optional fields within those messages and describes their behavior and interpretation. This architecture and protocol specification is independent of a specific wire encoding.

The protocol also uses a Control message called an InterestReturn, whereby one system can return an Interest message to the previous hop due to an error condition. This indicates to the previous hop that the current system will not respond to the Interest.

This document is a product of the Information Centric Networking research group (ICNRG).

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 http://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 July 28, 2019.

Table of Contents

1. Introduction

This document describes the principles of the CCNx architecture. It describes a network protocol that uses a hierarchical name to forward requests and to match responses to requests. It does not use endpoint addresses, such as Internet Protocol. Restrictions in a request can limit the response by the public key of the response's signer or the cryptographic hash of the response. Every CCNx forwarder along the path does the name matching and restriction checking. The CCNx protocol fits within the broader framework of Information Centric Networking (ICN) protocols [RFC7927]. This document concerns the semantics of the protocol and is not dependent on a specific wire format encoding. The CCNx Messages [CCNMessages] document describes a type-length-value (TLV) wire protocol encoding. This section introduces the main concepts of CCNx, which are further elaborated in the remainder of the document.

The CCNx protocol derives from the early ICN work by Jacobson et al. [nnc]. Jacobson's version of CCNx is known as the 0.x version ("CCNx 0.x") and the present work is known as the 1.0 version ("CCNx 1.0"). There are two active implementations of CCNx 1.0. The most complete implementation is Community ICN (CINC) [cicn], a Linux Foundation project hosted at fd.io. Another active implementation is CCN-lite [ccnlite], with support for IoT systems and the RIOT operating system. CCNx 0.x formed the basis of the Named Data Networking [ndn] (NDN) university project.

The current CCNx 1.0 specification diverges from CCNx 0.x in a few significant areas. The most pronounced behavioral difference between CCNx 0.x and CCNx 1.0 is that CCNx 1.0 has a simpler response processing behavior. In both versions, a forwarder uses a hierarchical longest prefix match of a request name against the forwarding information base (FIB) to send the request through the network to a system that can issue a response. A forwarder must then match a response's name to a request's name to determine the reverse

path and deliver the response to the requester. In CCNx 0.x, the Interest name may be a hierarchical prefix of the response name, which allows a form of layer 3 content discovery. In CCNx 1.0, a response's name must exactly equal a request's name. Content discovery is performed by a higher-layer protocol.

CCNx Selectors [selectors] is an example of using a higher-layer protocol on top of the CCNx 1.0 layer-3 to perform content discovery. The selector protocol uses a method similar to the original CCNx 0.x techniques without requiring partial name matching of a response to a request in the forwarder.

The document represents the consensus of the ICN RG. It is the first ICN protocol from the RG, created from the early CCNx protocol [nnc] with significant revision and input from the ICN community and RG members. The draft has received critical reading by several members of the ICN community and the RG. The authors and RG chairs approve of the contents. The document is sponsored under the IRTF and is not issued by the IETF and is not an IETF standard. This is an experimental protocol and may not be suitable for any specific application and the specification may change in the future.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.2. Architecture

We describe the architecture of the network in which CCNx operates and introduce certain terminology from [terminology]. The detailed behavior of each component and message grammars are in Section 2.

A producer (also called a publisher) is an endpoint that encapsualtes content in Content Objects for transport in the CCNx network. A producer has a public/private keypair and signs (directly or indirectly) the content objects. Usually, the producer's keyid (hash of the public key) is well-known or may be derived from the producer's namespace via standard means.

A producer operates within one or more namespaces. A namespace is a name prefix that is represented in the forwarding information base (FIB). This allows a request to reach the producer and fetch a response (if one exists).

The forwarding information base (FIB) is a table that tells a forwarder where to send a request. It may point to a local

application, a local cache or content store, or to a remote system. If there is no matching entry in the FIB, a forwarder cannot process a request. The detailed rules on name matching to the FIB are given in Section 2.4.4. An endpoint has a FIB, though it may be a simple default route. An intermediate system (i.e. a router) typically has a much larger FIB. A core CCNx forwarder, for example, would know all the global routes.

A consumer is an endpoint that requests a name. It is beyond the scope of this document to describe how a consumer learns of a name or publisher keyid -- higher layer protocols build on top of CCNx handle those tasks, such as search engines or lookup services or well known names. The consumer constructs a request, called an Interest, and forwards it via the endpoint's FIB. The consumer should get back either a response, called a Content Object, that matches the Interest or a control message, called an InterestReturn, that indicates the network cannot handle the request.

There are three ways to detect errors in Interest handling. An InterestReturn is a network control message that indicates a low-level error like no route or out of resources. If an Interest arrives at a producer, but the producer does not have the requested content, the producer should send an application-specific error message (e.g. a not found message). Finally, a consumer may not receive anything, in which case it should timeout and, depending on the application, retry the request or return an error to the application.

1.3. Protocol Overview

The goal of CCNx is to name content and retrieve the content from the network without binding it to a specific network endpoint. A routing system (specified separately) populates the forwarding information base (FIB) tables at each CCNx router with hierarchical name prefixes that point towards the content producers under that prefix. A request finds matching content along those paths, in which case a response carries the data, or if no match is found a control message indicates the failure. A request may further refine acceptable responses with a restriction on the response's signer and the cryptographic hash of the response. The details of these restrictions are described below.

The CCNx name is a hierarchical series of path segments. Each path segment has a type and zero or more bytes. Matching two names is done as a binary comparison of the type and value, segment by segment. The human-readable form is defined under a URI scheme "ccnx:" [CCNxURI], though the canonical encoding of a name is a series of (type, octet string) pairs. There is no requirement that

any path segment be human readable or UTF-8. The first few segments in a name will matched against the FIB and a routing protocol may put its own restrictions on the routable name components (e.g. a maximum length or character encoding rules). In principle, path segments and names have unbounded length, though in practice they are limited by the wire format encoding and practical considerations imposed by a routing protocol. Note that in CCNx path segments use binary comparison whereas in a URI the authority uses case-insensitive hostname (due to DNS).

The CCNx name, as used by the forwarder, is purposefully left as a general octet-encoded type and value without any requirements on human readability and character encoding. The reason for this is that we are concerned with how a forwarder processes names. We expect that applications, routing protocols, or other higher layers will apply their own conventions and restrictions on the allowed path segment types and path segment values.

CCNx is a request and response protocol to fetch chunks of data using a name. The integrity of each chunk may be directly asserted through a digital signature or Message Authentication Code (MAC), or, alternatively, indirectly via hash chains. Chunks may also carry weaker message integrity checks (MICs) or no integrity protection mechanism at all. Because provenance information is carried with each chunk (or larger indirectly protected block), we no longer need to rely on host identities, such as those derived from TLS certificates, to ascertain the chunk legitimacy. Data integrity is therefore a core feature of CCNx; it does not rely on the data transmission channel. There are several options for data confidentiality, discussed later.

This document only defines the general properties of CCNx names. In some isolated environments, CCNx users may be able to use any name they choose and either inject that name (or prefix) into a routing protocol or use other information foraging techniques. In the Internet environment, there will be policies around the formats of names and assignments of names to publishers, though those are not specified here.

The key concept of CCNx is that a subjective name is cryptographically bound to a fixed payload. These publisher-generated bindings can therefore be cryptographically verified. A named payload is thus the tuple {{Name, ExtraFields, Payload, ValidationAlgorithm}, ValidationPayload}, where all fields in the inner tuple are covered by the validation payload (e.g. signature). Consumers of this data can check the binding integrity by re-computing the same cryptographic hash and verifying the digital signature in ValidationPayload.

In addition to digital signatures (e.g. RSA), CCNx also supports message authentication codes (e.g. HMAC) and message integrity codes (e.g. SHA-256 or CRC). To maintain the cryptographic binding, there should be at least one object with a signature or authentication code, but not all objects require it. For example, a first object with a signature could refer to other objects via a hash chain, a Merkle tree, or a signed manifest. The later objects may not have any validation and rely purely on the references. The use of an integrity code (e.g. CRC) is intended for detecting accidental corruption in an Interest.

CCNx specifies a network protocol around Interests (request messages) and Content Objects (response messages) to move named payloads. An Interest includes the Name -- which identifies the desired response -- and optional matching restrictions. Restrictions limit the possible matching Content Objects. Two restrictions exist: KeyIdRestr and ContentObjectHashRestr. The first restriction on the KeyId limits responses to those signed with a ValidationAlgorithm KeyId field equal to the restriction. The second is the Content ObjectHash restriction, which limits the response to one where the cryptographic hash of the entire named payload is equal to the restriction.

The hierarchy of a CCNx Name is used for routing via the longest matching prefix in a Forwarder. The longest matching prefix is computed name segment by name segment in the hierarchical path name, where each name segment must be exactly equal to match. There is no requirement that the prefix be globally routable. Within a deployment any local routing may be used, even one that only uses a single flat (non-hierarchical) name segment.

Another concept of CCNx is that there should be flow balance between Interest messages and Content Object messages. At the network level, an Interest traveling along a single path should elicit no more than one Content Object response. If some node sends the Interest along more than one path, that node should consolidate the responses such that only one Content Object flows back towards the requester. If an Interest is sent broadcast or multicast on a multiple-access media, the sender should be prepared for multiple responses unless some other media-dependent mechanism like gossip suppression or leader election is used.

As an Interest travels the forward path following the Forwarding Information Base (FIB), it establishes state at each forwarder such that a Content Object response can trace its way back to the original requester(s) without the requester needing to include a routable return address. We use the notional Pending Interest Table (PIT) as

a method to store state that facilitates the return of a Content Object.

The notional PIT table stores the last hop of an Interest plus its Name and optional restrictions. This is the data required to match a Content Object to an Interest (see Section 9). When a Content Object arrives, it must be matched against the PIT to determine which entries it satisfies. For each such entry, at most one copy of the Content Object is sent to each listed last hop in the PIT entries.

An actual PIT table is not mandated by the specification. An implementation may use any technique that gives the same external behavior. There are, for example, research papers that use techniques like label switching in some parts of the network to reduce the per-node state incurred by the PIT table [dart]. Some implementations store the PIT state in the FIB, so there is not a second table.

If multiple Interests with the same {Name, KeyIdRestr, ContentObjectHashRestr} tuple arrive at a node before a Content Object matching the first Interest comes back, they are grouped in the same PIT entry and their last hops aggregated (see Section 2.4.2). Thus, one Content Object might satisfy multiple pending Interests in a PIT.

In CCNx, higher-layer protocols are often called "name-based protocols" because they operate on the CCNx Name. For example, a versioning protocol might append additional name segments to convey state about the version of payload. A content discovery protocol might append certain protocol-specific name segments to a prefix to discover content under that prefix. Many such protocols may exist and apply their own rules to Names. They may be layered with each protocol encapsulating (to the left) a higher layer's Name prefix.

This document also describes a control message called an InterestReturn. A network element may return an Interest message to a previous hop if there is an error processing the Interest. The returned Interest may be further processed at the previous hop or returned towards the Interest origin. When a node returns an Interest it indicates that the previous hop should not expect a response from that node for the Interest, i.e., there is no PIT entry left at the returning node for a Content Object to follow.

There are multiple ways to describe larger objects in CCNx. Aggregating layer-3 content objects in to larger objects is beyond the scope of this document. One proposed method, FLIC [flic], uses a manifest to enumerate the pieces of a larger object. Manifests are, themselves, Content Objects. Another option is to use a convention

in the Content Object name, as in the CCNx Chunking [chunking] protocol where a large object is broken in to small chunks and each chunk receives a special name component indicating its serial order.

At the semantic level, described in this document, we do not address fragmentation. One experimental fragmentation protocol, BeginEnd Fragments [befrags] uses a multipoint-PPP style technique for use over layer-2 interfaces with the CCNx Messages [CCNMessages] TLV wire forman specification.

With these concepts, the remainder of the document specifies the behavior of a forwarder in processing Interest, Content Object, and InterestReturn messages.

2. Protocol

CCNx is a request and response protocol. A request is called an Interest and a response is called a Content Object. CCNx also uses a 1-hop control message called InterestReturn. These are, as a group, called CCNx Messages.

2.1. Message Grammar

The CCNx message ABNF [RFC5234] grammar is shown in Figure 1. The grammar does not include any encoding delimiters, such as TLVs. Specific wire encodings are given in a separate document. If a Validation section exists, the Validation Algorithm covers from the Body (BodyName or BodyOptName) through the end of the ValidationAlg section. The InterestLifetime, CacheTime, and Return Code fields exist outside of the validation envelope and may be modified.

The various fields -- in alphabetical order -- are defined as:

- AbsTime: Absolute times are conveyed as the 64-bit UTC time in milliseconds since the epoch (standard POSIX time).

- CacheTime: The absolute time after which the publisher believes there is low value in caching the content object. This is a recommendation to caches (see Section 4).

- ConObjField: These are optional fields that may appear in a Content Object.

- ConObjHash: The value of the Content Object Hash, which is the SHA256-32 over the message from the beginning of the body to the end of the message. Note that this coverage area is different from the ValidationAlg. This value SHOULD NOT be trusted across domains (see Section 5).

- ExpiryTime: An absolute time after which the content object should be considered expired (see Section 4).

- Hash: Hash values carried in a Message carry a HashType to identify the algorithm used to generate the hash followed by the hash value. This form is to allow hash agility. Some fields may mandate a specific HashType.

- HopLimit: Interest messages may loop if there are loops in the forwarding plane. To eventually terminate loops, each Interest carries a HopLimit that is decremented after each hop and no longer forwarded when it reaches zero. See Section 2.4.

- InterestField: These are optional fields that may appear in an Interest message.

- KeyIdRestr: The KeyId Restriction. A Content Object must have a KeyId with the same value as the restriction.

- ContentObjectHashRestr: The Content Object Hash Restriction. A content object must hash to the same value as the restriction using the same HashType. The ContentObjectHashRestr MUST use SHA256-32.

- KeyId: An identifier for the key used in the ValidationAlg. For public key systems, this should be the SHA-256 hash of the public key. For symmetric key systems, it should be an identifier agreed upon by the parties.

- KeyLink: A Link (see Section 6) that names how to retrieve the key used to verify the ValidationPayload. A message SHOULD NOT have both a KeyLink and a PublicKey.

- Lifetime: The approximate time during which a requester is willing to wait for a response, usually measured in seconds. It is not strongly related to the network round trip time, though it must necessarily be larger.

- Name: A name is made up of a non-empty first segment followed by zero or more additional segments, which may be of 0 length. Path segments are opaque octet strings, and are thus case-sensitive if encoding UTF-8. An Interest MUST have a Name. A Content Object MAY have a Name (see Section 9). The segments of a name are said to be complete if its segments uniquely identify a single Content Object. A name is exact if its segments are complete. An Interest carrying a full name is one which specifies an exact name and the ContentObjectHashRestr of the corresponding Content Object.

- Payload: The message's data, as defined by PayloadType.

- PayloadType: The format of the Payload. If missing, assume DataType. DataType means the payload is opaque application bytes. KeyType means the payload is a DER-encoded public key. LinkType means it is one or more Links (see Section 6).

- PublicKey: Some applications may wish to embed the public key used to verify the signature within the message itself. The PublickKey is DER encoded. A message SHOULD NOT have both a KeyLink and a PublicKey.

- RelTime: A relative time, measured in milli-seconds.

- ReturnCode: States the reason an Interest message is being returned to the previous hop (see Section 10.2).

- SigTime: The absolute time (UTC milliseconds) when the signature was generated.

- Vendor: Vendor-specific opaque data. The Vendor data includes the IANA Private Enterprise Numbers [EpriseNumbers], followed by vendor-specific information. CCNx allows vendor-specific data in most locations of the grammar.

```
Message       := Interest / ContentObject / InterestReturn
Interest      := IntHdr BodyName [Validation]
IntHdr        := HopLimit [Lifetime] *Vendor
ContentObject := ConObjHdr BodyOptName [Validation]
ConObjHdr     := [CacheTime / ConObjHash] *Vendor
InterestReturn:= ReturnCode Interest
BodyName      := Name Common
BodyOptName   := [Name] Common
Common        := *Field [Payload]
Validation    := ValidationAlg ValidatonPayload

Name          := FirstSegment *Segment
FirstSegment  := 1* OCTET / Vendor
Segment       := 0* OCTET / Vendor

ValidationAlg := (RSA-SHA256 / HMAC-SHA256 / CRC32C) *Vendor
ValidatonPayload := 1* OCTET
RSA-SHA256    := KeyId [PublicKey] [SigTime] [KeyLink]
HMAC-SHA256   := KeyId [SigTime] [KeyLink]
CRC32C        := [SigTime]

AbsTime       := 8 OCTET ; 64-bit UTC msec since epoch
CacheTime     := AbsTime
```

```
ConObjField   := ExpiryTime / PayloadType
ConObjHash    := Hash ; The Content Object Hash
DataType      := "1"
ExpiryTime    := AbsTime
Field         := InterestField / ConObjField / Vendor
Hash          := HashType 1* OCTET
HashType      := SHA256-32 / SHA512-64 / SHA512-32
HopLimit      := OCTET
InterestField := KeyIdRestr / ContentObjectHashRestr
KeyId         := 1* OCTET ; key identifier
KeyIdRestr    := 1* OCTET
KeyLink       := Link
KeyType       := "2"
Lifetime      := RelTime
Link          := Name [KeyIdResr] [ContentObjectHashRestr]
LinkType      := "3"
ContentObjectHashRestr  := Hash
Payload       := *OCTET
PayloadType   := DataType / KeyType / LinkType
PublicKey     := ; DER-encoded public key
RelTime       := 1* OCTET ; msec
ReturnCode    := ; see Section 10.2
SigTime       := AbsTime
Vendor        := PEN 0* OCTET
PEN           := ; IANA Private Enterprise Number
```

Figure 1

2.2. Consumer Behavior

To request a piece of content for a given {Name, [KeyIdRest], [ContentObjectHashRestr]} tuple, a consumer creates an Interest message with those values. It MAY add a validation section, typically only a CRC32C. A consumer MAY put a Payload field in an Interest to send additional data to the producer beyond what is in the Name. The Name is used for routing and may be remembered at each hop in the notional PIT table to facilitate returning a content object; Storing large amounts of state in the Name could lead to high memory requirements. Because the Payload is not considered when forwarding an Interest or matching a Content Object to an Interest, a consumer SHOULD put an Interest Payload ID (see Section 3.2) as part of the name to allow a forwarder to match Interests to content objects and avoid aggregating Interests with different payloads. Similarly, if a consumer uses a MAC or a signature, it SHOULD also include a unique segment as part of the name to prevent the Interest from being aggregated with other Interests or satisfied by a Content Object that has no relation to the validation.

The consumer SHOULD specify an InterestLifetime, which is the length of time the consumer is willing to wait for a response. The InterestLifetime is an application-scale time, not a network round trip time (see Section 2.4.2). If not present, the InterestLifetime will use a default value (2 seconds).

The consumer SHOULD set the Interest HopLimit to a reasonable value or use the default 255. If the consumer knows the distances to the producer via routing, it SHOULD use that value.

A consumer hands off the Interest to its first forwarder, which will then forward the Interest over the network to a publisher (or replica) that may satisfy it based on the name (see Section 2.4).

Interest messages are unreliable. A consumer SHOULD run a transport protocol that will retry the Interest if it goes unanswered, up to the InterestLifetime. No transport protocol is specified in this document.

The network MAY send to the consumer an InterestReturn message that indicates the network cannot fulfill the Interest. The ReturnCode specifies the reason for the failure, such as no route or congestion. Depending on the ReturnCode, the consumer MAY retry the Interest or MAY return an error to the requesting application.

If the content was found and returned by the first forwarder, the consumer will receive a Content Object. The consumer SHOULD:

- Ensure the content object is properly formatted.
- Verify that the returned Name matches a pending request. If the request also had KeyIdRestr or ObjHashRest, it MUST also validate those properties.
- If the content object is signed, it SHOULD cryptographically verify the signature. If it does not have the corresponding key, it SHOULD fetch the key, such as from a key resolution service or via the KeyLink.
- If the signature has a SigTime, the consumer MAY use that in considering if the signature is valid. For example, if the consumer is asking for dynamically generated content, it should expect the SigTime to not be before the time the Interest was generated.
- If the content object is signed, it should assert the trustworthiness of the signing key to the namespace. Such an assertion is beyond the scope of this document, though one may use

traditional PKI methods, a trusted key resolution service, or methods like [trust].

- It MAY cache the content object for future use, up to the ExpiryTime if present.

- A consumer MAY accept a content object off the wire that is expired. It may happen that a packet expires while in flight, and there is no requirement that forwarders drop expired packets in flight. The only requirement is that content stores, caches, or producers MUST NOT respond with an expired content object.

2.3. Publisher Behavior

This document does not specify the method by which names populate a Forwarding Information Base (FIB) table at forwarders (see Section 2.4). A publisher is either configured with one or more name prefixes under which it may create content, or it chooses its name prefixes and informs the routing layer to advertise those prefixes.

When a publisher receives an Interest, it SHOULD:

- Verify that the Interest is part of the publishers namespace(s).

- If the Interest has a Validation section, verify the ValidationPayload. Usually an Interest will only have a CRC32C unless the publisher application specifically accommodates other validations. The publisher MAY choose to drop Interests that carry a Validation section if the publisher application does not expect those signatures as this could be a form of computational denial of service. If the signature requires a key that the publisher does not have, it is NOT RECOMMENDED that the publisher fetch the key over the network, unless it is part of the application's expected behavior.

- Retrieve or generate the requested content object and return it to the Interest's previous hop. If the requested content cannot be returned, the publisher SHOULD reply with an InterestReturn or a content object with application payload that says the content is not available; this content object should have a short ExpiryTime in the future or not be cacheable (i.e. an expiry time of 0).

2.4. Forwarder Behavior

A forwarder routes Interest messages based on a Forwarding Information Base (FIB), returns Content Objects that match Interests to the Interest's previous hop, and processes InterestReturn control messages. It may also keep a cache of Content Objects in the

notional Content Store table. This document does not specify the internal behavior of a forwarder -- only these and other external behaviors.

In this document, we will use two processing pipelines, one for Interests and one for Content Objects. Interest processing is made up of checking for duplicate Interests in the PIT (see Section 2.4.2), checking for a cached Content Object in the Content Store (see Section 2.4.3), and forwarding an Interest via the FIB. Content Store processing is made up of checking for matching Interests in the PIT and forwarding to those previous hops.

2.4.1. Interest HopLimit

Interest looping is not prevented in CCNx. An Interest traversing loops is eventually discarded using the hop-limit field of the Interest, which is decremented at each hop traversed by the Interest.

A loop may also terminate because the Interest is aggregated with it's previous PIT entry along the loop. In this case, the Content will be sent back along the loop and eventually return to a node that already forwarded the content, so it will likely not have a PIT entry any more. When the content reaches a node without a PIT entry, it will be discarded. It may be that a new Interest or another looped Interest will return to that same node, in which case the node will either return a cached response to make a new PIT entry, as below.

The HopLimit is the last resort method to stop Interest loops where a Content Object chases an Interest around a loop and where the intermediate nodes, for whatever reason, no longer have a PIT entry and do not cache the Content Object.

Every Interest MUST carry a HopLimit. An Interest received from a local application MAY have a 0 HopLimit, which restricts the Interest to other local sources.

When an Interest is received from another forwarder, the HopLimit MUST be positive, otherwise the forwarder will discard the Interest. A forwarder MUST decrement the HopLimit of an Interest by at least 1 before it is forwarded.

If the decremented HopLimit equals 0, the Interest MUST NOT be forwarded to another forwarder; it MAY be sent to a local publisher application or serviced from a local Content Store.

A RECOMMENDED HopLimit processing pipeline is below:

- If Interest received from a remote system:

    * If received HopLimit is 0, optionally send InterestReturn (HopLimit Exceeded), and discard Interest.

    * Otherwise, decrement the HopLimit by 1.

- Process as per Content Store and Aggregation rules.

- If the Interest will be forwarded:

    * If the (potentailly decremented) HopLimit is 0, restrict forwarding to the local system.

    * Otherwise, forward as desired to local or remote systems.

2.4.2. Interest Aggregation

Interest aggregation is when a forwarder receives an Interest message that could be satisfied by the response to another Interest message already forwarded by the node, so the forwarder suppresses forwarding the new Interest; it only records the additional previous hop so a Content Object sent in response to the first Interest will satisfy both Interests.

CCNx uses an Interest aggregation rule that assumes the InterestLifetime is akin to a subscription time and is not a network round trip time. Some previous aggregation rules assumed the lifetime was a round trip time, but this leads to problems of expiring an Interest before a response comes if the RTT is estimated too short or interfering with an ARQ scheme that wants to re-transmit an Interest but a prior interest over-estimated the RTT.

A forwarder MAY implement an Interest aggregation scheme. If it does not, then it will forward all Interest messages. This does not imply that multiple, possibly identical, Content Objects will come back. A forwarder MUST still satisfy all pending Interests, so one Content Object could satisfy multiple similar interests, even if the forwarded did not suppress duplicate Interest messages.

A RECOMMENDED Interest aggregation scheme is:

- Two Interests are considered 'similar' if they have the same Name, KeyIdRestr, and ContentObjectHashRestr.

- Let the notional value InterestExpiry (a local value at the forwarder) be equal to the receive time plus the InterestLifetime (or a platform-dependent default value if not present).

- An Interest record (PIT entry) is considered invalid if its InterestExpiry time is in the past.

- The first reception of an Interest MUST be forwarded.

- A second or later reception of an Interest similar to a valid pending Interest from the same previous hop MUST be forwarded. We consider these a retransmission requests.

- A second or later reception of an Interest similar to a valid pending Interest from a new previous hop MAY be aggregated (not forwarded). If this Interest has a larger HopLimit than the pending Interest, it MUST be forwarded.

- Aggregating an Interest MUST extend the InterestExpiry time of the Interest record. An implementation MAY keep a single InterestExpiry time for all previous hops or MAY keep the InterestExpiry time per previous hop. In the first case, the forwarder might send a Content Object down a path that is no longer waiting for it, in which case the previous hop (next hop of the Content Object) would drop it.

2.4.3. Content Store Behavior

The Content Store is a special cache that is an integral part of a CCNx forwarder. It is an optional component. It serves to repair lost packets and handle flash requests for popular content. It could be pre-populated or use opportunistic caching. Because the Content Store could serve to amplify an attack via cache poisoning, there are special rules about how a Content Store behaves.

1. A forwarder MAY implement a Content Store. If it does, the Content Store matches a Content Object to an Interest via the normal matching rules (see Section 9).

2. If an Interest has a KeyIdRestr, then the Content Store MUST NOT reply unless it knows the signature on the matching Content Object is correct. It may do this by external knowledge (i.e., in a managed network or system with pre-populated caches) or by having the public key and cryptographically verifying the signature. A Content Store is NOT REQURIED to verify signatures; if it does not, then it treats these cases like a cache miss.

3. If a Content Store chooses to verify signatures, then it MAY do so as follows. If the public key is provided in the Content Object itself (i.e., in the PublicKey field) or in the Interest, the Content Store MUST verify that the public key's SHA-256 hash is equal to the KeyId and that it verifies the signature. A

Content Store MAY verify the digital signature of a Content Object before it is cached, but it is not required to do so. A Content Store SHOULD NOT fetch keys over the network. If it cannot or has not yet verified the signature, it should treat the Interest as a cache miss.

4. If an Interest has an ContentObjectHashRestr, then the Content Store MUST NOT reply unless it knows the the matching Content Object has the correct hash. If it cannot verify the hash, then it should treat the Interest as a cache miss.

5. It must obey the Cache Control directives (see Section 4).

2.4.4. Interest Pipeline

1. Perform the HopLimit check (see Section 2.4.1).

2. Determine if the Interest can be aggregated, as per Section 2.4.2. If it can be, aggregate and do not forward the Interest.

3. If forwarding the Interest, check for a hit in the Content Store, as per Section 2.4.3. If a matching Content Object is found, return it to the Interest's previous hop. This injects the Content Store as per Section 2.4.5.

4. Lookup the Interest in the FIB. Longest prefix match (LPM) is performed name segment by name segment (not byte or bit). It SHOULD exclude the Interest's previous hop. If a match is found, forward the Interest. If no match is found or the forwarder choses to not forward due to a local condition (e.g., congestion), it SHOULD send an InterestReturn message, as per Section 10.

2.4.5. Content Object Pipeline

1. It is RECOMMENDED that a forwarder that receives a content object check that the Content Object came from an expected previous hop. An expected previous hop is one pointed to by the FIB or one recorded in the PIT as having had a matching Interest sent that way.

2. A Content Object MUST be matched to all pending Interests that satisfy the matching rules (see Section 9). Each satisfied pending Interest MUST then be removed from the set of pending Interests.

3. A forwarder SHOULD NOT send more then one copy of the received Content Object to the same Interest previous hop. It may happen, for example, that two Interest ask for the same Content Object in different ways (e.g., by name and by name an KeyId) and that they both come from the same previous hop. It is normal to send the same content object multiple times on the same interface, such as Ethernet, if it is going to different previous hops.

4. A Content Object SHOULD only be put in the Content Store if it satisfied an Interest (and passed rule #1 above). This is to reduce the chances of cache poisoning.

3. Names

A CCNx name is a composition of name segments. Each name segment carries a label identifying the purpose of the name segment, and a value. For example, some name segments are general names and some serve specific purposes, such as carrying version information or the sequencing of many chunks of a large object into smaller, signed Content Objects.

There are three different types of names in CCNx: prefix, exact, and full names. A prefix name is simply a name that does not uniquely identify a single Content Object, but rather a namespace or prefix of an existing Content Object name. An exact name is one which uniquely identifies the name of a Content Object. A full name is one which is exact and is accompanied by an explicit or implicit ConObjHash. The ConObjHash is explicit in an Interest and implicit in a Content Object.

Note that a forwarder does not need to know any semantics about a name. It only needs to be able to match a prefix to forward Interests and match an exact or full name to forward Content Objects. It is not sensitive to the name segment types.

The name segment labels specified in this document are given in the table below. Name Segment is a general name segment, typically occurring in the routable prefix and user-specified content name. Other segment types are for functional name components that imply a specific purpose.

Name	Description
Name Segment	A generic name segment that includes arbitrary octets.
Interest Payload ID	An octet string that identifies the payload carried in an Interest. As an example, the Payload ID might be a hash of the Interest Payload. This provides a way to differentiate between Interests based on the Payload solely through a Name Segment without having to include all the extra bytes of the payload itself.
Application Components	An application-specific payload in a name segment. An application may apply its own semantics to these components. A good practice is to identify the application in a Name segment prior to the application component segments.

Table 1: CCNx Name Segment Types

At the lowest level, a Forwarder does not need to understand the semantics of name segments; it need only identify name segment boundaries and be able to compare two name segments (both label and value) for equality. The Forwarder matches paths segment-by-segment against its forwarding table to determine a next hop.

3.1. Name Examples

This section uses the CCNx URI [CCNxURI] representation of CCNx names. Note that as per the message grammar, an Interest must have a Name with at least one name segment and that name segment must have at least 1 octet of value. A Content Object must have a similar name or no name at all. The FIB, on the other hand, could have 0-length names (a default route), or a first name segment with no value, or a regular name.

Name	Description
ccnx:/	A 0-length name, corresponds to a default route.
ccnx:/NAME=	A name with 1 segment of 0 length, distinct from ccnx:/.
ccnx:/NAME=foo/APP:0=bar	A 2-segment name, where the first segment is of type NAME and the second segment is of type APP:0.

Table 2: CCNx Name Examples

3.2. Interest Payload ID

An Interest may also have a Payload which carries state about the Interest but is not used to match a Content Object. If an Interest contains a payload, the Interest name should contain an Interest Payload ID (IPID). The IPID allows a PIT table entry to correctly multiplex Content Objects in response to a specific Interest with a specific payload ID. The IPID could be derived from a hash of the payload or could be a GUID or a nonce. An optional Metadata field defines the IPID field so other systems could verify the IPID, such as when it is derived from a hash of the payload. No system is required to verify the IPID.

4. Cache Control

CCNx supports two fields that affect cache control. These determine how a cache or Content Store handles a Content Object. They are not used in the fast path, but only to determine if a Content Object can be injected on to the fast path in response to an Interest.

The ExpiryTime is a field that exists within the signature envelope of a Validation Algorithm. It is the UTC time in milliseconds after which the Content Object is considered expired and MUST no longer be used to respond to an Interest from a cache. Stale content MAY be flushed from the cache.

The Recommended Cache Time (RCT) is a field that exists outside the signature envelope. It is the UTC time in milliseconds after which the publisher considers the Content Object to be of low value to cache. A cache SHOULD discard it after the RCT, though it MAY keep it and still respond with it. A cache MAY discard the content object

before the RCT time too; there is no contractual obligation to remember anything.

This formulation allows a producer to create a Content Object with a long ExpiryTime but short RCT and keep re-publishing the same, signed, Content Object over and over again by extending the RCT. This allows a form of "phone home" where the publisher wants to periodically see that the content is being used.

5. Content Object Hash

CCNx allows an Interest to restrict a response to a specific hash. The hash covers the Content Object message body and the validation sections, if present. Thus, if a Content Object is signed, its hash includes that signature value. The hash does not include the fixed or hop-by-hop headers of a Content Object. Because it is part of the matching rules (see Section 9), the hash is used at every hop.

There are two options for matching the content object hash restriction in an Interest. First, a forwarder could compute for itself the hash value and compare it to the restriction. This is an expensive operation. The second option is for a border device to compute the hash once and place the value in a header (ConObjHash) that is carried through the network. The second option, of course, removes any security properties from matching the hash, so SHOULD only be used within a trusted domain. The header SHOULD be removed when crossing a trust boundary.

6. Link

A Link is the tuple {Name, [KeyIdRestr], [ContentObjectHashRestr]}. The information in a Link comprises the fields of an Interest which would retrieve the Link target. A Content Object with PayloadType = "Link" is an object whose payload is one or more Links. This tuple may be used as a KeyLink to identify a specific object with the certificate wrapped key. It is RECOMMENDED to include at least one of KeyIdRestr or Content ObjectHashRestr. If neither restriction is present, then any Content Object with a matching name from any publisher could be returned.

7. Hashes

Several protocol fields use cryptographic hash functions, which must be secure against attack and collisions. Because these hash functions change over time, with better ones appearing and old ones falling victim to attacks, it is important that a CCNx protocol implementation supports hash agility.

In this document, we suggest certain hashes (e.g., SHA-256), but a specific implementation may use what it deems best. The normative CCNx Messages [CCNMessages] specification should be taken as the definition of acceptable hash functions and uses.

8. Validation

8.1. Validation Algorithm

The Validator consists of a ValidationAlgorithm that specifies how to verify the message and a ValidationPayload containing the validation output, e.g., the digital signature or MAC. The ValidationAlgorithm section defines the type of algorithm to use and includes any necessary additional information. The validation is calculated from the beginning of the CCNx Message through the end of the ValidationAlgorithm section. The ValidationPayload is the integrity value bytes, such as a MAC or signature.

Some Validators contain a KeyId, identifying the publisher authenticating the Content Object. If an Interest carries a KeyIdRestr, then that KeyIdRestr MUST exactly match the Content Object's KeyId.

Validation Algorithms fall into three categories: MICs, MACs, and Signatures. Validators using Message Integrity Code (MIC) algorithms do not need to provide any additional information; they may be computed and verified based only on the algorithm (e.g., CRC32C). MAC validators require the use of a KeyId identifying the secret key used by the authenticator. Because MACs are usually used between two parties that have already exchanged secret keys via a key exchange protocol, the KeyId may be any agreed-upon value to identify which key is used. Signature validators use public key cryptographic algorithms such as RSA, DSA, ECDSA. The KeyId field in the ValidationAlgorithm identifies the public key used to verify the signature. A signature may optionally include a KeyLocator, as described above, to bundle a Key or Certificate or KeyLink. MAC and Signature validators may also include a SignatureTime, as described above.

A PublicKeyLocator KeyLink points to a Content Object with a DER-encoded X509 certificate in the payload. In this case, the target KeyId must equal the first object's KeyId. The target KeyLocator must include the public key corresponding to the KeyId. That key must validate the target Signature. The payload is an X.509 certificate whose public key must match the target KeyLocator's key. It must be issued by a trusted authority, preferably specifying the valid namespace of the key in the distinguished name.

9. Interest to Content Object matching

A Content Object satisfies an Interest if and only if (a) the Content Object name, if present, exactly matches the Interest name, and (b) the ValidationAlgorithm KeyId of the Content Object exactly equals the Interest KeyIdRestr, if present, and (c) the computed Content ObjectHash exactly equals the Interest ContentObjectHashRestr, if present.

The matching rules are given by this predicate, which if it evaluates true means the Content Object matches the Interest. Ni = Name in Interest (may not be empty), Ki = KeyIdRestr in the interest (may be empty), Hi = ContentObjectHashRestr in Interest (may be empty). Likewise, No, Ko, Ho are those properties in the Content Object, where No and Ko may be empty; Ho always exists (it is an intrinsic property of the Content Object). For binary relations, we use & for AND and | for OR. We use E for the EXISTS (not empty) operator and ! for the NOT EXISTS operator.

As a special case, if the ContentObjectHashRestr in the Interest specifies an unsupported hash algorithm, then no Content Object can match the Interest so the system should drop the Interest and MAY send an InterestReturn to the previous hop. In this case, the predicate below will never get executed because the Interest is never forwarded. If the system is using the optional behavior of having a different system calculate the hash for it, then the system may assume all hash functions are supported and leave it to the other system to accept or reject the Interest.

```
(!No | (Ni=No)) & (!Ki | (Ki=Ko)) & (!Hi | (Hi=Ho)) & (E No | E Hi)
```

As one can see, there are two types of attributes one can match. The first term depends on the existence of the attribute in the Content Object while the next two terms depend on the existence of the attribute in the Interest. The last term is the "Nameless Object" restriction which states that if a Content Object does not have a Name, then it must match the Interest on at least the Hash restriction.

If a Content Object does not carry the Content ObjectHash as an expressed field, it must be calculated in network to match against. It is sufficient within an autonomous system to calculate a Content ObjectHash at a border router and carry it via trusted means within the autonomous system. If a Content Object ValidationAlgorithm does not have a KeyId then the Content Object cannot match an Interest with a KeyIdRestr.

10. Interest Return

This section describes the process whereby a network element may return an Interest message to a previous hop if there is an error processing the Interest. The returned Interest may be further processed at the previous hop or returned towards the Interest origin. When a node returns an Interest it indicates that the previous hop should not expect a response from that node for the Interest -- i.e., there is no PIT entry left at the returning node.

The returned message maintains compatibility with the existing TLV packet format (a fixed header, optional hop-by-hop headers, and the CCNx message body). The returned Interest packet is modified in only two ways:

- The PacketType is set to InterestReturn to indicate a Feedback message.
- The ReturnCode is set to the appropriate value to signal the reason for the return

The specific encodings of the Interest Return are specified in [CCNMessages].

A Forwarder is not required to send any Interest Return messages.

A Forwarder is not required to process any received Interest Return message. If a Forwarder does not process Interest Return messages, it SHOULD silently drop them.

The Interest Return message does not apply to a Content Object or any other message type.

An Interest Return message is a 1-hop message between peers. It is not propagated multiple hops via the FIB. An intermediate node that receives an InterestReturn may take corrective actions or may propagate its own InterestReturn to previous hops as indicated in the reverse path of a PIT entry.

10.1. Message Format

The Interest Return message looks exactly like the original Interest message with the exception of the two modifications mentioned above. The PacketType is set to indicate the message is an InterestReturn and the reserved byte in the Interest header is used as a Return Code. The numeric values for the PacketType and ReturnCodes are in [CCNMessages].

10.2. ReturnCode Types

This section defines the InterestReturn ReturnCode introduced in this RFC. The numeric values used in the packet are defined in [CCNMessages].

Name	Description
No Route (Section 10.3.1)	The returning Forwarder has no route to the Interest name.
HopLimit Exceeded (Section 10.3.2)	The HopLimit has decremented to 0 and need to forward the packet.
Interest MTU too large (Section 10.3.3)	The Interest's MTU does not conform to the required minimum and would require fragmentation.
No Resources (Section 10.3.4)	The node does not have the resources to process the Interest.
Path error (Section 10.3.5)	There was a transmission error when forwarding the Interest along a route (a transient error).
Prohibited (Section 10.3.6)	An administrative setting prohibits processing this Interest.
Congestion (Section 10.3.7)	The Interest was dropped due to congestion (a transient error).
Unsupported Content Object Hash Algorithm (Section 10.3.8)	The Interest was dropped because it requested a Content Object Hash Restriction using a hash algorithm that cannot be computed.
Malformed Interest (Section 10.3.9)	The Interest was dropped because it did not correctly parse.

Table 3: Interest Return Reason Codes

10.3. Interest Return Protocol

This section describes the Forwarder behavior for the various Reason codes for Interest Return. A Forwarder is not required to generate

any of the codes, but if it does, it MUST conform to this specification.

If a Forwarder receives an Interest Return, it SHOULD take these standard corrective actions. A forwarder is allowed to ignore Interest Return messages, in which case its PIT entry would go through normal timeout processes.

- Verify that the Interest Return came from a next-hop to which it actually sent the Interest.
- If a PIT entry for the corresponding Interest does not exist, the Forwarder should ignore the Interest Return.
- If a PIT entry for the corresponding Interest does exist, the Forwarder MAY do one of the following:
  - Try a different forwarding path, if one exists, and discard the Interest Return, or
  - Clear the PIT state and send an Interest Return along the reverse path.

If a forwarder tries alternate routes, it MUST ensure that it does not use same same path multiple times. For example, it could keep track of which next hops it has tried and not re-use them.

If a forwarder tries an alternate route, it may receive a second InterestReturn, possibly of a different type than the first InterestReturn. For example, node A sends an Interest to node B, which sends a No Route return. Node A then tries node C, which sends a Prohibited. Node A should choose what it thinks is the appropriate code to send back to its previous hop

If a forwarder tries an alternate route, it should decrement the Interest Lifetime to account for the time spent thus far processing the Interest.

10.3.1. No Route

If a Forwarder receives an Interest for which it has no route, or for which the only route is back towards the system that sent the Interest, the Forwarder SHOULD generate a "No Route" Interest Return message.

How a forwarder manages the FIB table when it receives a No Route message is implementation dependent. In general, receiving a No Route Interest Return should not cause a forwarder to remove a route.

The dynamic routing protocol that installed the route should correct the route or the administrator who created a static route should correct the configuration. A forwarder could suppress using that next hop for some period of time.

10.3.2. HopLimit Exceeded

A Forwarder MAY choose to send HopLimit Exceeded messages when it receives an Interest that must be forwarded off system and the HopLimit is 0.

10.3.3. Interest MTU Too Large

If a Forwarder receives an Interest whose MTU exceeds the prescribed minimum, it MAY send an "Interest MTU Too Large" message, or it may silently discard the Interest.

If a Forwarder receives an "Interest MTU Too Large" is SHOULD NOT try alternate paths. It SHOULD propagate the Interest Return to its previous hops.

10.3.4. No Resources

If a Forwarder receives an Interest and it cannot process the Interest due to lack of resources, it MAY send an InterestReturn. A lack of resources could be the PIT table is too large, or some other capacity limit.

10.3.5. Path Error

If a forwarder detects an error forwarding an Interest, such as over a reliable link, it MAY send a Path Error Interest Return indicating that it was not able to send or repair a forwarding error.

10.3.6. Prohibited

A forwarder may have administrative policies, such as access control lists, that prohibit receiving or forwarding an Interest. If a forwarder discards an Interest due to a policy, it MAY send a Prohibited InterestReturn to the previous hop. For example, if there is an ACL that says /parc/private can only come from interface e0, but the Forwarder receives one from e1, the Forwarder must have a way to return the Interest with an explanation.

10.3.7. Congestion

If a forwarder discards an Interest due to congestion, it MAY send a Congestion InterestReturn to the previous hop.

10.3.8. Unsupported Content Object Hash Algorithm

If a Content Object Hash Restriction specifies a hash algorithm the forwarder cannot verify, the Interest should not be accepted and the forwarder MAY send an InterestReturn to the previous hop.

10.3.9. Malformed Interest

If a forwarder detects a structural or syntactical error in an Interest, it SHOULD drop the interest and MAY send an InterestReturn to the previous hop. This does not imply that any router must validate the entire structure of an Interest.

11. IANA Considerations

This memo includes no request to IANA.

12. Security Considerations

The CCNx protocol is a layer 3 network protocol, which may also operate as an overlay using other transports, such as UDP or other tunnels. It includes intrinsic support for message authentication via a signature (e.g. RSA or elliptic curve) or message authentication code (e.g. HMAC). In lieu of an authenticator, it may instead use a message integrity check (e.g. SHA or CRC). CCNx does not specify an encryption envelope, that function is left to a high-layer protocol (e.g. [esic]).

The CCNx message format includes the ability to attach MICs (e.g. SHA-256 or CRC), MACs (e.g. HMAC), and Signatures (e.g. RSA or ECDSA) to all packet types. This does not mean that it is a good idea to use an arbitrary ValidationAlgorithm, nor to include computationally expensive algorithms in Interest packets, as that could lead to computational DoS attacks. Applications should use an explicit protocol to guide their use of packet signatures. As a general guideline, an application might use a MIC on an Interest to detect unintentionally corrupted packets. If one wishes to secure an Interest, one should consider using an encrypted wrapper and a protocol that prevents replay attacks, especially if the Interest is being used as an actuator. Simply using an authentication code or signature does not make an Interests secure. There are several examples in the literature on how to secure ICN-style messaging [mobile] [ace].

As a layer 3 protocol, this document does not describe how one arrives at keys or how one trusts keys. The CCNx content object may include a public key embedded in the object or may use the PublicKeyLocator field to point to a public key (or public key certificate) that authenticates the message. One key exchange specification is CCNxKE [ccnxke] [mobile], which is similar to the TLS 1.3 key exchange except it is over the CCNx layer 3 messages. Trust is beyond the scope of a layer-3 protocol protocol and left to applications or application frameworks.

The combination of an ephemeral key exchange (e.g. CCNxKE [ccnxke]) and an encapsulating encryption (e.g. [esic]) provides the equivalent of a TLS tunnel. Intermediate nodes may forward the Interests and Content Objects, but have no visibility inside. It also completely hides the internal names in those used by the encryption layer. This type of tunneling encryption is useful for content that has little or no cache-ability as it can only be used by someone with the ephemeral key. Short term caching may help with lossy links or mobility, but long term caching is usually not of interest.

Broadcast encryption or proxy re-encryption may be useful for content with multiple uses over time or many consumers. There is currently no recommendation for this form of encryption.

The specific encoding of messages will have security implications. [CCNMessages] uses a type-length-value (TLV) encoding. We chose to compromise between extensibility and unambiguous encodings of types and lengths. Some TLVs use variable length T and variable length L fields to accomodate a wide gamut of values while trying to be byte-efficient. Our TLV encoding uses a fixed length 2-byte T and 2-byte L. Using a fixed-length T and L field solves two problems. The first is aliases. If one is able to encode the same value, such as 0x2 and 0x02, in different byte lengths then one must decide if they mean the same thing, if they are different, or if one is illegal. If they are different, then one must always compare on the buffers not the integer equivalents. If one is illegal, then one must validate the TLV encoding -- every field of every packet at every hop. If they are the same, then one has the second problem: how to specify packet filters. For example, if a name has 6 name components, then there are 7 T's and 7 L's, each of which might have up to 4 representations of the same value. That would be 14 fields with 4 encodings each, or 1001 combinations. It also means that one cannot compare, for example, a name via a memory function as one needs to consider that any embedded T or L might have a different format.

The Interest Return message has no authenticator from the previous hop. Therefore, the payload of the Interest Return should only be used locally to match an Interest. A node should never forward that

Interest payload as an Interest. It should also verify that it sent the Interest in the Interest Return to that node and not allow anyone to negate Interest messages.

Caching nodes must take caution when processing content objects. It is essential that the Content Store obey the rules outlined in Section 2.4.3 to avoid certain types of attacks. Unlike NDN, CCNx 1.0 has no mechanism to work around an undesired result from the network (there are no "excludes"), so if a cache becomes poisoned with bad content it might cause problems retrieving content. There are three types of access to content from a content store: unrestricted, signature restricted, and hash restricted. If an Interest has no restrictions, then the requester is not particular about what they get back, so any matching cached object is OK. In the hash restricted case, the requester is very specific about what they want and the content store (and every forward hop) can easily verify that the content matches the request. In the signature verified case (often used for initial manifest discovery), the requester only knows the KeyId that signed the content. It is this case that requires the closest attention in the content store to avoid amplifying bad data. The content store must only respond with a content object if it can verify the signature -- this means either the content object carries the public key inside it or the Interest carries the public key in addition to the KeyId. If that is not the case, then the content store should treat the Interest as a cache miss and let an endpoint respond.

A user-level cache could perform full signature verification by fetching a public key according to the PublicKeyLocator. That is not, however, a burden we wish to impose on the forwarder. A user-level cache could also rely on out-of-band attestation, such as the cache operator only inserting content that it knows has the correct signature.

The CCNx grammar allows for hash algorithm agility via the HashType. It specifies a short list of acceptable hash algorithms that should be implemented at each forwarder. Some hash values only apply to end systems, so updating the hash algorithm does not affect forwarders -- they would simply match the buffer that includes the type-length-hash buffer. Some fields, such as the ConObjHash, must be verified at each hop, so a forwarder (or related system) must know the hash algorithm and it could cause backward compatibility problems if the hash type is updated. [CCNMessages] is the authoritative source for per-field allowed hash types in that encoding.

A CCNx name uses binary matching whereas a URI uses a case insensitive hostname. Some systems may also use case insensitive matching of the URI path to a resource. An implication of this is

that human-entered CCNx names will likely have case or non-ASCII symbol mismatches unless one uses a consistent URI normalization to the CCNx name. It also means that an entity that registers a CCNx routable prefix, say ccnx:/example.com, would need separate registrations for simple variations like ccnx:/Example.com. Unless this is addressed in URI normalization and routing protocol conventions, there could be phishing attacks.

For a more general introduction to ICN-related security concerns and approaches, see [RFC7927] and [RFC7945]

13. References

13.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

13.2. Informative References

[ace] Shang, W., Yu, Y., Liang, T., Zhang, B., and L. Zhang, "NDN-ACE: Access control for constrained environments over named data networking", NDN Technical Report NDN-0036, 2015, <http://new.named-data.net/wp-content/uploads/2015/12/ndn-0036-1-ndn-ace.pdf>.

[befrags] Mosko, M. and C. Tschudin, "ICN "Begin-End" Hop by Hop Fragmentation", 2017, <https://www.ietf.org/archive/id/draft-mosko-icnrg-beginendfragment-02.txt>.

[ccnlite] Tschudin, C., et al., University of Basel, "CCN-Lite V2", 2011-2018, <http://www.ccn-lite.net/>.

[CCNMessages] Mosko, M., Solis, I., and C. Wood, "CCNx Messages in TLV Format (Internet draft)", 2018, <https://www.ietf.org/id/draft-irtf-icnrg-ccnxmessages-07.txt>.

[ccnxke] Mosko, M., Uzun, E., and C. Wood, "CCNx Key Exchange Protocol Version 1.0", 2017, <https://www.ietf.org/archive/id/draft-wood-icnrg-ccnxkeyexchange-02.txt>.

[CCNxURI] Mosko, M. and C. Wood, "The CCNx URI Scheme (Internet draft)", 2017, <http://tools.ietf.org/html/draft-mosko-icnrg-ccnxuri-02>.

[chunking]
Mosko, M., "CCNx Content Object Chunking", 2016, <https://www.ietf.org/archive/id/draft-mosko-icnrg-ccnxchunking-02.txt>.

[cicn] Muscariello, L., et al., Cisco Systems, "Community ICN (CICN)", 2017-2018, <https://wiki.fd.io/view/Cicn>.

[dart] Garcia-Luna-Aceves, J. and M. Mirzazad-Barijough, "A Light-Weight Forwarding Plane for Content-Centric Networks", 2016, <https://arxiv.org/pdf/1603.06044.pdf>.

[EpriseNumbers]
IANA, "IANA Private Enterprise Numbers", 2015, <http://www.iana.org/assignments/enterprise-numbers/enterprise-numbers>.

[esic] Mosko, M. and C. Wood, "Encrypted Sessions In CCNx (ESIC)", 2017, <https://www.ietf.org/id/draft-wood-icnrg-esic-01.txt>.

[flic] Tschudin, C. and C. Wood, "File-Like ICN Collection (FLIC)", 2017, <https://www.ietf.org/archive/id/draft-tschudin-icnrg-flic-03.txt>.

[mobile] Mosko, M., Uzun, E., and C. Wood, "Mobile Sessions in Content-Centric Networks", IFIP Networking, 2017, <http://dl.ifip.org/db/conf/networking/networking2017/1570334964.pdf>.

[ndn] UCLA, "Named Data Networking", 2007, <http://www.named-data.net>.

[nnc] Jacobson, V., Smetters, D., Thornton, J., Plass, M., Briggs, N., and R. Braynard, "Networking Named Content", 2009, <http://dx.doi.org/10.1145/1658939.1658941>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC7927] Kutscher, D., Eum, S., Pentikousis, K., Psaras, I., Corujo, D., Saucez, D., Schmidt, T., and M. Waehlisch, "Information-Centric Networking (ICN) Research Challenges", 2016, <https://trac.tools.ietf.org/html/rfc7927>.

[RFC7945] Pentikousis, K., Ohlman, B., Davies, E., Spirou, S., and G. Boggia, "Information-Centric Networking: Evaluation and Security Considerations", 2016, <https://trac.tools.ietf.org/html/rfc7945>.

[selectors] Mosko, M., "CCNx Selector Based Discovery", 2017, <https://raw.githubusercontent.com/mmosko/ccnx-protocol-rfc/master/docs/build/draft-mosko-icnrg-selectors-01.txt>.

[terminology] Wissingh, B., Wood, C., Afanasyev, A., Zhang, L., Oran, D., and C. Tschudin, "Information-Centric Networking (ICN): CCN and NDN Terminology", 2017, <https://www.ietf.org/id/draft-irtf-icnrg-terminology-00.txt>.

[trust] Tschudin, C., Uzun, E., and C. Wood, "Trust in Information-Centric Networking: From Theory to Practice", 2016, <https://doi.org/10.1109/ICCCN.2016.7568589>.

Authors' Addresses

Marc Mosko
PARC, Inc.
Palo Alto, California 94304
USA

Phone: +01 650-812-4405
Email: marc.mosko@parc.com

Ignacio Solis
LinkedIn
Mountain View, California 94043
USA

Email: nsolis@linkedin.com

Christopher A. Wood
University of California Irvine
Irvine, California 92697
USA

Phone: +01 315-806-5939
Email: woodc1@uci.edu

ICNRG
Internet-Draft
Intended status: Informational
Expires: September 20, 2016

D. Kutscher, Ed.
NEC
S. Eum
NICT
K. Pentikousis
EICT
I. Psaras
UCL
D. Corujo
Universidade de Aveiro
D. Saucez
INRIA
T. Schmidt
HAW HAmburg
M. Waehlisch
FU Berlin
March 19, 2016

ICN Research Challenges
draft-irtf-icnrg-challenges-06

Abstract

This memo describes research challenges for Information-Centric Networking (ICN), an approach to evolve the Internet infrastructure to directly support information distribution by introducing uniquely named data as a core Internet principle. Data becomes independent from location, application, storage, and means of transportation, enabling or enhancing a number of desirable features, such as security, user-mobility, multicast and in-network caching. Mechanisms for realizing these benefits is subject of on-going research in the IRTF and elsewhere. This document describes current research challenges in ICN, including naming, security, routing, system scalability, mobility management, wireless networking, transport services, in-network caching, and network management.

This document is a product of the IRTF Information-Centric Networking Research Group (ICNRG).

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 http://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 20, 2016.

Table of Contents

1. Introduction

Information-centric networking (ICN) is an approach to evolve the Internet infrastructure to directly support this use by introducing named data as a network primitive. Data objects become independent of location, application, storage, and means of transportation, allowing for inexpensive and ubiquitous in-network caching and replication. The expected benefits are improved efficiency and security, better scalability with respect to information/bandwidth demand and better robustness in challenging communication scenarios.

ICN concepts can be deployed by retooling the protocol stack: name-based data access can be implemented on top of the existing IP infrastructure, e.g., by allowing for named data structures, ubiquitous caching and corresponding transport services, or it can be seen as a packet-level internetworking technology that would cause fundamental changes to Internet routing and forwarding. In summary, ICN can evolve the Internet architecture towards a named-data-based network model with different properties and different services.

This document presents the ICN research challenges that need to be addressed in order to achieve these goals. These research challenges are seen from a technical perspective, although business relationships between Internet players will also influence developments in this area. We leave business challenges for a separate document, however. The objective of this memo is to document the technical challenges and corresponding current

approaches and to expose requirements that should be addressed by future research work.

This document has been reviewed, commented, and discussed extensively for a period of nearly two years by the vast majority of ICNRG members, which certainly exceeds 100 individuals. It is the consensus of ICNRG that the research challenges described in this document should be published in the IRTF Stream of the RFC series. This document does not constitute a standard.

2. Problems with Host-Centric Communications

The best current practice to manage the above-mentioned growth in terms of data volume and number of devices is to increase infrastructure investment, employ application-layer overlays that cache content such as Content Distribution Networks (CDNs) and Peer-to-Peer (P2P) applications, provide location-independent access to data, and optimize its delivery. In principle, such platforms provide a service model of accessing named data objects (NDOs) (e.g., replicated web resources in data centers) instead of a host-to-host packet delivery service model.

However, since this functionality resides in overlays only, the full potential of content distribution platforms cannot be leveraged as the network is not aware of data requests and data transmissions. This has the following impact:

- data traffic typically follows sub-optimal paths as it is effectively routed depending on the overlay topology instead of the Internet layer topology;
- network capabilities, such as multicast and broadcast, are largely underutilized or not employed at all. As a result, request and delivery for the same object have to be made multiple times;
- overlays typically require significant infrastructure support, e.g., authentication portals, content storage, and applications servers, making it often impossible to establish local, direct communication;
- the forwarding layer cannot cooperate with transport layer functions,so sometimes useful functionality such as local retransmission, local rate control have to implemented with TCP proxies or other intermediaries.
- provenance validation uses host authentication today. As such, even if there are locally cached copies available, it is normally not easily possible to validate their authenticity; and

- o many applications follow their own approach to caching, replication, transport, authenticity validation (if at all), although they all share similar models for accessing named data objects in the network.

Host-centric communication systems restrict applications to data transfer between end-hosts only. Naming data directly provides a powerful "hook" for applications to exploit and natively support multi-party communication, e.g., multi-source/multi-destination communication and a ubiquitous information ecosystem that is not restricted to end-host addresses.

3. ICN Terminology and Concepts

3.1. Terminology

Information-Centric Networking (ICN): A concept for communicating in a network that provides accessing named data objects as a first order service. See Section 3.2 for details.

Named Data Object (NDO): Addressable data unit in an information-centric network that can represent a collection of bytes or a piece of information. In ICN, each data object has a name bound to it, and there are typically mechanisms to secure (and validate) this binding. Different ICN approaches have different concepts for how to map NDOs to individual units of transport, e.g., chunks, segments. Sometimes smaller units may be represented by NDOs themselves. Within the context of this document, an NDO is any named data object that can be requested from the network, and we do not consider sub units below the NDO level. In this document we often use the terms NDO and data object interchangeably.

Requestor: Entity in an ICN network that is sending a request for a Named Data Object to the network.

Publisher: Entity in an ICN network that publishes an NDO to the network, so that corresponding requests can reach the publisher. The publisher does not need to be identical to the actual creator, for example a publisher could provide the service of hosting NDOs on behalf of the actual creators/owners.

3.2. Concepts

Fundamentally, ICN provides access to named data as a first-order network service, i.e., the network is able to serve requests to named data natively. That means, network nodes can receive requests for named data and act as necessary, for example, by forwarding the

request to a suitable next-hop. Consequently, the network processes requests for named data objects (and corresponding responses) natively. Every network node on a path is enabled to perform forwarding decisions, cache objects etc. This enables the network to forward such requests on optimal paths, employing the best transmission technologies at every node, e.g., broadcast/multicast transmission in wireless networks to avoid duplicate transmission of both requests and responses.

In ICN there is a set of common concepts and node requirements beyond this basic service model. Naming data objects is a key concept. In general, ICN names represent neither network nodes nor interfaces -- they represent NDOs independently of their location. Names do play a key role in forwarding decisions and are used for matching requests to responses: In order to provide better support for accessing copies of NDOs regardless of their location, it is important to be able to validate that a response actually delivers the bits that correspond to an original request for named data.

Name-content binding validation is a fundamental security service in ICN, and this is often achieved by establishing a verifiable binding between the object name and the actual object or an identity that has created the object. ICN can support other security services, such as provenance validation and encryption depending on the details of naming schemes, object models and assumptions on infrastructure support. Security services such as name-content binding validation are available to any node, i.e., not just the actual requestors. This is an important feature, for enabling ingress gateways to check object authenticity to prevent denial-of-service attacks.

Based on these fundamental properties it is possible to leverage network storage ubiquitously: every ICN node can cache data objects and respond to requests for such objects -- it is not required to validate the authenticity of the node itself since name-content bindings can be validated. Ubiquitous in-network storage can be used for different purposes: it can enable sharing, i.e., the same object copy can be delivered to multiple users/nodes as in today’s proxy caches and CDNs. It can also be used to make communication more robust (and perform better) by enabling the network to answer requests from local caches (instead of from origin servers). In case of disruption (message not delivered), a node can resend the request, and it could be answered by an on-path cache, i.e., on the other side of the disrupted link. The network itself would thus support retransmissions enabling shorter round-trip times and offloading origin servers and other parts of the network.

ICN potentially retrieves segments of NDOs from multiple data sources, and so only a requestor can determine the completion of a

retrieval process, i.e., the retrieval of NDOs or individual segments is typically controlled by a requestor. For this reason, ICN transport protocols are typically based on a receiver-driven mechanism: requestors can control message sending rates by regulating the request sending rate (assuming that every response message has to be triggered by a request message). Retransmission would be achieved by resending requests, e.g., after a timeout. Because objects can be replicated, object transmission and transport sessions would not necessarily have end-to-end semantics: requests can be answered by caches, and a node can select one or multiple next-hop destinations for a particular request depending on configuration, observed performance or other criteria.

This receiver-driven communication model potentially enables new interconnection and business models: a request for named data can be linked to an interest of a requestor (or requesting network) in data from another peer, which could suggest modeling peering agreements and charging accordingly.

4. ICN Research Challenges

4.1. Naming, Data Integrity, and Data Orign Authentication

Naming data objects is as important for ICN as naming hosts is for today's Internet. Fundamentally, ICN requires unique names for individual NDOs, since names are used for identifying objects independently of their location or container. In addition, since NDOs can be cached anywhere, the origin cannot be trusted anymore hence the importance to establish a verifiable binding between the object and its name (name-data binding validation) so that a requestor can be sure that the received bits do correspond to the NDO originally requested (data integrity). Data orign authentication is a different security service that can be related to naming, i.e., verifying that an NDO has indeed been published by a publisher (that could be identified by a name prefix).

The above functions are fundamentally required for the information-centric network to work reliably, otherwise neither network elements nor requestors can trust object authenticity. Lack of this trust enables several attacks including DoS attacks by injecting spoofed content into the network. There are different ways to use names and cryptography to achieve the desired functions [ICNNAMING] [ICNSURVEY], and there are different ways to manage namespaces correspondingly.

Two types of naming schemes have been proposed in the ICN literature: hierarchical and flat namespaces. For example, a hierarchical scheme may have a structure similar to current URIs, where the hierarchy is

rooted in a publisher prefix. Such hierarchy enables aggregation of routing information, improving scalability of the routing system. In some cases, names are human-readable, which makes it possible for users to manually type in names, reuse and, to some extent, map the name to user intent.

The second general class of naming schemes enables verifying the object's name-data integrity without requiring a public key infrastructure (PKI) or other third party to first establish trust in the key. This is achieved, e.g., by binding the hash of the NDO content to the object's name. For instance, this can be done by directly embedding the hash of the content in the name. Another option is an indirect binding, which embeds the public key of the publisher in the name and signs the hash of the content with the corresponding private key. The resulting names are typically non-hierarchical, or flat, although the publisher field could be employed to create a structure which could facilitate route aggregation.

There are several design trade-offs for ICN naming, which affect routing and security. Hash-based names are not human readable nor hierarchical. They can however provide some structure for aggregation, for instance, a name part corresponding to a publisher. In hash-based names with indirect binding, the key of the publisher is bound to the name of NDO, and so when a user receives, e.g., the triplet namely [data, key, signature], the receiving entity can verify that the NDO has been generated by the possessor of the private/public key pair and that the NDO has not been changed in transit (data integrity). This can be done by cryptographically hashing the received key and the name of NDO, and comparing it with received hashed key. Then, the key can be used to verify the signature.

Data origin authentication can be achieved by validating public-key-cryptography-based signatures about an NDO's name and content. In order to ascertain data integrity and origin authenticity with such an approach, a PKI-like system is required that would allow linking the corresponding public key to a trust chain.

Research challenges specific to naming include:

- Naming static data objects can be performed by using content hashes as part of object names, so that publishers can calculate the hash over existing data objects and requestors and any ICN node can validate the name-content binding by re-calculating the hash and comparing it to the name (component). [RFC6920] specifies a concrete naming format for this.

- Naming dynamic objects refers to use cases where the name has to be generated before the object is created. For example, this could be the case for live streaming, when a publisher wants to make the stream available by registering stream chunk names in the network. One approach to this can be hash-based names with indirect binding as described above.

- Requestor privacy protection can be a challenge in ICN as a direct consequence of the accessing-named-data-objects paradigm: if the network can "see" requests and responses, it can also log request history for network segments or individual users, which can be undesirable, especially since names are typically expected to be long-lived. That is, even if the name itself does not reveal much information, the assumption is that the name can be used to retrieve the corresponding data objects in the future.

- Updating and versioning NDOs can be challenging because it can contradict fundamental ICN assumptions: if an NDO can be replicated and stored in in-network storage for later retrieval, names have to be long-lived and the name-content binding must not change: updating an object (i.e., changing the content without generating a new name) is not possible. Versioning is one possible solution, but requires a naming scheme that supports it (and a way for requestors to learn about newer and older versions).

- Managing accessibility: whereas in ICN the general assumption is to enable ubiquitous access to NDOs, there can be relevant use cases where access to objects should be restricted, for example to a specific user group. There are different approaches for this, such as object encryption (requiring key distribution and related mechanisms) or the concept of scopes, e.g., based on names that can only be used/resolved under some constraints.

4.2. Security

Security is an active research field in ICN. This section provides an overview of important important security features and corresponding challenges that are related to shifting to information-centric communications. Some challenges are well-understood, and there are (sometimes multiple different) approaches to address them, whereas other challenges are active research and engineering topics.

4.2.1. Data Integrity and Orign Authentication

As mentioned in section Section 4.1, data integrity verification is an important ICN feature, since NDOs are retrieved not only from an original copy holder but also from any caching point. Hence, the

communication channel endpoints to retrieve NDOs are not trustable anymore and solutions widely used today such as TLS [RFC5246] cannot be used as a general solution. Since data objects can be maliciously modified, ICN should provide receivers with a security mechanism to verify the integrity of the data object, and there are different ways to achieve this.

An efficient approach for static NDOs is providing a name-content-binding by hashing an NDO and using the hash as a part of the object's name. [RFC6920] provides a mechanism and a format for representing a digest algorithm and the actual digest in a name (amongst other information).

For dynamic objects where it is desirable to refer to an NDO by name before the object has been created, public-key cryptography is often applied, i.e., every NDO would be authenticated by means of a signature performed by the data object publisher so that any data object consumer can verify the validity of the data object based on the signature. However, in order to verify the signature of an object, the consumer must know the public key of the entity that signed the object.

Data Orign Authentication, i.e., verifying that an NDO has indeed been published by a publisher, requires a secure binding of an NDO name to a publisher identiy -- this is also typically implemented using public key cryptography, i.e., by requiring a recevier to verify digital signatures that as part of a received message.

One research challenge is then to support a mechanism to distribute the publisher's public keys to the consumers of data objects. There are two main approaches to achieve this; one is based on an external third party authority such as hierarchical Public Key Infrastructure (PKI) (see [RFC5280] for a description of hierarchical PKI) and the other is to adapt a hash-based scheme with indirect binding. The former, as the name implies, depends on an external third party authority to distribute the public key of the publisher for the consumers. In a hash-based scheme with indirect binding, the public key (or a hash of it) would be used as part of the name -- which is sufficient to validate the data integrity.

In cases where information about the origin of a data object is not available by other means, the object itself would have to incorporate the necessary information to determine the object publisher, for example with a certificate, that can be validated through the PKI. Once the certificate is authenticated, its public key can be used to authenticate the signed data object itself.

4.2.2. Binding NDOs to Real-World Identities

In addition to validating NDO authenticity, it is still important to bind real-world identities, e.g., a publisher identity, to objects, so that a requestor can verify that a received object was actually published by a certain source.

With hash-based names, real world identity bindings are not intrinsically established: the name provides the hash of the NDO or of the public key that has been used to sign the NDO. There needs to be another binding to a real world identity if that feature is requested.

If the object name directly provides the publisher name and if that name is protected by a certificate that links to a PKI-like trust chain, the object name itself can provide an intrinsic binding to a real world identity.

Binding between NDOs and real world identities is essential but there is no universal way to achieve it as it is all intrinsic to a particular ICN approach.

4.2.3. Access Control and Authorization

Access control and authorization is a challenge in ICN, because of the lack of user-to-server authentication in the fundamental named-data-based communication model.

All ICN entities are capable of delivering NDOs on demand due to their in-network caching function. In such an environment, traditional access control schemes based on Access Control List (ACL) are ill-suited since widely distributed ICN entities have to maintain an identical control policy over NDOs for each consumer, which is prohibited due to computational overhead and privacy issues. There are two complementary approaches to address the issues:

1. Separated approach: access control service from a third party that is independent from ICN entities. Due to the clear separation, ICN entities free from computational overhead to determine the accessibility of NDOs by consumers and also consumers can secure their privacy through the independent authorization entity [access-control-delegation]. Relevant challenges to this approach include reducing the authorization delay (when communicating to the access control provider) and currency and consistency of access control information (when access control lists are distributed).

2. Integrated approach: access control service from ICN entities. This mechanism is often based on content encryption and key distribution [encryption-ac]. As mentioned previously, this approach suffers from prohibitive overhead for ICN entities due to the process of key verification. While key distribution is per se challenging, this approach is beneficial in a way that NDOs can be retrieved without the help of an external access control provider. Challenges to this approach include:

    1. applying an access control mechanism for dynamic NDOs in in-network caches in a timely manner;

    2. providing consumers with the different levels of accessibility to individual NDOs in a scalable manner; and

    3. managing key revocation and similar PKI management functions.

4.2.4. Encryption

In ICN, NDOs can be encrypted to implement access control (only consumers in possession of corresponding decryption keys can access the content) or privacy (same approach). Distributing and managing the corresponding keys as well as providing usable interfaces to applications and human users is a challenge and subject of on-going work.

In principle, the challenges are similar to those of broadcast/media distribution, and similar approaches (combing symmetric with public-key cryptography) are being investigated. [ndn-controlled-sharing]

4.2.5. Traffic Aggregation and Filtering

One request message to retrieve a data object can actually aggregate requests coming from several consumers. This aggregation of requests reduces the overall traffic but makes per-requestor filtering harder. The challenge in this case is to provide a mechanism that allows request aggregation and per-requestor filtering. A possible solution is to indicate the set of requestors in the aggregated request such that the response can indicate the subset of requestors allowed to access the data object. However, this solution requires collaboration from other nodes in the network and is not suitable for caching. Another possible solution is to encrypt data objects and ensure that only authorised consumers can decrypt them. This solution does not preclude caching and does not require collaboration from the network. However, it implies a mechanism to generate group keys (e.g., different private keys can be used to decrypt the same encrypted data object) [Chaum].

4.2.6. State Overloading

ICN solutions that implement state on intermediate routers for request routing or forwarding (e.g., CCN [CCN]) are subject to denial of service attacks from overloading or superseding the internal state of a router (e.g., 'interest flooding' [BACKSCATTER]). Additionally, stateful forwarding can enable attack vectors such as resource exhaustion or complexity attacks to the routing infrastructure. The challenge is then to provision routers and construct internal state in a way that alleviates sensibility to such attacks. The problem becomes even harder, if the protocol does not provide information about the origin of messages. Without origin, it is a particular challenge to distinguish between regular (intense) use and misuse of the infrastructure.

4.2.7. Delivering Data Objects from Replicas

A common capability of ICN solutions is data replication and in-network storage. Delivering replicated data objects from caches decouples content consumption from data sources, which leads to a loss of control on (1) content access, and (2) content dissemination. In a widely distributed, decentralized environment like the Internet, this raises several challenges.

One group of challenges is related to content management. Without access control, a content provider loses the means to count and survey content consumption, to limit access scopes, to control or know about the number of copies of its data in the network, or to withdraw earlier publications reliably. Any non-cooperative or desynchronized data cache may hinder an effective content management policy.

Another group of challenges arises from potential traffic amplifications in the decoupled environment. ICN solutions that attempt to retrieve content from several replicas in parallel, or decorrelated network routing states, but also distributed attackers may simultaneously initiate the transmission of content from multiple replicas towards the same destination (e.g., 'initiated overloads' or 'blockades' [BACKSCATTER]). Methods for mitigating such threats need rigorous forwarding checks that require alignment with caching procedures (e.g., on-path or off-path).

4.2.8. Cryptographic Robustness

Content producers sign their content to ensure the integrity of data and to allow for data object authentication. This is a fundamental requirement in ICN due to distributed caching. Publishers, who (a) massively sign content, which is (b) long-lived, offer time and data

to an attacker for comprising cryptographic credentials. Signing large amount of data eases common attacks that try to breach the key of the publisher. Based on this observation, the following research challenges emerge:

- To which extent does the content publication model conflict with cryptographic limitations?
- How can we achieve transparent re-signing without introducing additional cryptographic weaknesses or key management overhead?

In general, ICN implementations should be designed considering the guidelines provided by [RFC7696], especially regarding cryptographic algorithm agility, for example [RFC6920] specifies a naming scheme for hash-based names that has been designed to support algorithm agility.

4.2.9. Routing and Forwarding Information Bases

In information-centric networks, one attack vector is to increase the size of routing and forwarding information bases at ICN nodes, i.e., attacking routing scalability in networks that rely on routing by name. This is an intrinsic ICN security issue: possible mitigation approaches include combining routing information authenticity validation with filtering (e.g., maximum de-aggregation level whenever applicable, black lists, etc.).

4.3. Routing and Resolution System Scalability

ICN routing is a process that finds a NDO based on its name initially provided by a requestor. ICN routing may comprise three steps: (i) name resolution, (ii) discovery, and (iii) delivery. The name resolution step translates the name of the requested NDO into its locator. The discovery step routes the request to data object based on its name or locator. The last step (delivery) routes the data object back to the requestor. Depending on how these steps are combined, ICN routing schemes can be categorized as Route-By-Name Routing (RBNR), Lookup-By-Name Routing (LBNR), and Hybrid Routing (HR) as discussed in the following subsections.

4.3.1. Route-By-Name Routing

RBNR omits the first name resolution step as the name of the NDO is directly used to route the request to the data object. Therefore, routing information for each data object has to be maintained in the routing table. Since the number of data objects is very large (estimated as 10^11 back in 2007 [DONA] but this may be significantly larger than that, e.g., 10^15 to 10^22), the size of routing tables

becomes a concern, as it can be proportional to the number of data objects unless an aggregation mechanism is introduced. On the other hand, RBNR reduces overall latency and simplifies the routing process due to the omission of the resolution process. For the delivery step, RBNR needs another identifier (ID) of either host or location to forward the requested data object back to the requestor. Otherwise, an additional routing mechanism has to be introduced, such as bread-crumbs routing [BREADCRUMBS], in which each request leaves behind a trail of breadcrumbs along its forwarding path, and then the response is forwarded back to the requestor consuming the trail.

Challenges specific to RBNR include:

- How can we aggregate the names of data objects to reduce the number of routing entries?

- How does a user learn the name which is designed for aggregation by provider? For example, although we name our contribution as "ICN research challenges", the IRTF (provider) may want to change the name to "/IETF/IRTF/ICN/Research challenges" for aggregation. In this case, how does a user learn the name "/IETF/IRTF/ICN/ Research challenges" to retrieve the contribution initially named "ICN research challenges" without any resolution process?

- Without introducing the name aggregation scheme, can we still achieve scalable routing by taking advantage of topological structure and distributed copies? For example, would employing compact routing [COMPACT], random walk [RANDOM] or greedy routing [GREEDY] work at Internet scale?

- How can we incorporate copies of a data object in in-network caches in this routing scheme?

- Bread-crumbs routing implies a symmetric path for ICN request and response delivery. Some network configurations and link types prohibit symmetric path forwarding, so it would be challenging to interconnect such networks to bread-crumbs routing-based infrastructure. For example, certain forwarding strategies in Delay-Tolerant Networking (DTN) [RFC4838] are employing opportunistic forwarding where responses cannot be assumed to travel the same path as requests.

4.3.2. Lookup-By-Name Routing

LBNR uses the first name resolution step to translate the name of the requesting data object into its locator. Then, the second discovery step is carried out based on the locator. Since IP addresses could be used as locators, the discovery step can depend on the current IP

infrastructure. The delivery step can be implemented similarly to IP routing. The locator of the requestor is included in the request message, and then the requested data object is delivered to the requestor based on the locator. An instantiation of LBNR is [MDHT].

Challenges specific to LBNR include:

- How can we build a scalable resolution system which provides
  - Fast lookup: mapping the name of data object to its locators (copies as well).
  - Fast update: the location of data object is expected to change frequently. Also, multiple data objects may change their locations at the same time, e.g., data objects in a laptop.
- How can we incorporate copies of a data object in in-network caches in this routing scheme?

4.3.3. Hybrid Routing

HR combines RBNR and LBNR to benefit from their advantages. Within a single administrative domain, e.g., an ISP, where scalability issues can be addressed with network planning, RBNR can be adopted to reduce overall latency by omitting the resolution process. On the other hand, LBNR can be used to route between domains which have their own prefix (locator).

For instance, a request message initially includes the name of NDO for the operation of RBNR, and is forwarded to a cached copy of the NDO or the original server. When the request message fails to find a routing entry in the router, a name resolution step kicks in to translate the name into its locator before forwarding the request message based on the retrieved locator.

Challenge specific to HR are:

- How can we design a scalable mapping system which, given the name of NDO, should return a destination domain locator so that a user request can be encapsulated and forwarded to the domain?
- How to secure the mapping information to prevent a malicious router from hijacking the request message by chaining its locator?
- How to maintain the bind between the name and the content of NDO for the verification of its origin and integrity when the name changes due to the retrieved locator?

4.4. Mobility Management

Mobility management has been an active field in host-centric communications for more than two decades. In IETF in particular, starting with [RFC2002], a multitude of enhancements to IP have been standardized aiming to "allow transparent routing of IP datagrams to mobile nodes in the Internet" [RFC5944]. In a nutshell [MMIN], mobility management for IP networks is locator-oriented and relies on the concept of a mobility anchor as a foundation for providing always-on connectivity to mobile nodes. Other standards organizations, such as 3GPP, have followed similar anchor-based approaches. Traffic to and from the mobile node must flow through the mobility anchor, typically using a set of tunnels, enabling the mobile node to remain reachable while changing its point of attachment to the network.

Needless to say, an IP network which supports node mobility is more complex than one that does not, as specialized network entities must be introduced in the network architecture. This is reflected in the control plane as well, which carries mobility-related signaling messages, establishes and tears down tunnels and so on. While mobile connectivity was an afterthought in IP, in ICN this is considered a primary deployment environment. Most, if not all, ICN proposals consider mobility from the very beginning, although at varying levels of architectural and protocol detail. That said, no solution has so far come forward with a definite answer on how to handle mobility in ICN using native primitives. In fact, we observe that mobility appears to be addressed on an ICN proposal-specific basis. That is, there is no single paradigm solution, akin to tunneling through a mobility anchor in host-centric networking, that can be applied across different ICN proposals. For instance, although widely-deployed mobile network architectures typically come with their own network entities and associated protocols, they follow the same line of design with respect to managing mobility. This design thinking, which calls for incorporating mobility anchors, permeates in the ICN literature too.

However, employing mobility anchors and tunneling is probably not the best way forward in ICN research for mobile networking. Fundamentally this approach is anything but information-centric and location-independent. In addition, as argued in [SEEN], current mobility management schemes anchor information retrieval not only at a specific network gateway (e.g., home agent in Mobile IP) but due to the end-to-end nature of host-centric communication also at a specific correspondent node. However, once a change in the point of attachment occurs, information retrieval from the original "correspondent node" may be no longer optimal. This was shown in [MANI], for example, where a simple mechanism that triggers the

discovery of new retrieval providers for the same data object, following a change in the point of attachment, clearly outperforms a tunnel-based approach like Mobile IP in terms of object download times. The challenge here is how to capitalize on location information while facilitating the use of ICN primitives which natively support multicast and anycast.

ICN naming and name resolution, as well as the security features that come along should natively support mobility. For example, CCN [CCN] does not have the restriction of spanning tree routing, so it is able to take advantage of multiple interfaces or adapt to the changes produced by rapid mobility (i.e., there is no need to bind a layer 3 address with a layer 2 address). In fact, client mobility can be simplified by allowing requests for new content to normally flow from different interfaces, or through newly connected points of attachment to the network. However, when the node moving is the (only) content source, it appears that more complex network support might be necessary, including forwarding updates and cache rebuilding. A case in point is a conversation network service, such as a voice or video call between two parties. The requirements in this case are more stringent when support for seamless mobility is required, especially when compared to content dissemination that is amenable to buffering. Another parameter that needs to be paid attention to is the impact of using different wireless access interfaces based on different technologies, where the performance and link conditions can vary widely depending of numerous factors.

In host-centric networking, mobility management mechanisms ensure optimal handovers and (ideally) seamless transition from one point of attachment to another. In ICN, however, the traditional meaning of "point of attachment" no longer applies as communication is not restrained by location-based access to data objects. Therefore, a "seamless transition" in ICN ensures that content reception continues without any perceptible change from the point of view of the ICN application receiving that content. Moreover, this transition needs to be executed in parallel with ICN content identification and delivery mechanisms enabling scenarios, such as, preparation of the content delivery process at the target connectivity point, prior to the handover (to reduce link switch disturbances). Finally, these mobility aspects can also be tightly coupled with network management aspects, in respect to policy enforcement, link control and other parameters necessary for establishing the node’s link to the network.

In summary, the following research challenges on ICN mobility management can be derived:

- How can mobility management take full advantage of native ICN primitives?

- How do we avoid the need for mobility anchors in a network that by design supports multicast, anycast and location-indepedent information retrieval?

- How can content retrieval mechanisms interface with specific link operations, such as identifying which links are available for certain content?

- How can mobility be offered as a service, which is only activated when the specific user/content/conditions require it?

- How can mobility management be coordinated between the node and the network for optimization and policing procedures?

- How do we ensure that managing mobility does not introduce scalability issues in ICN?

- How will the name resolution process be affected by rapid topological changes, when the content source itself is mobile?

4.5. Wireless Networking

Today, all layer 2 (L2) wireless network radio access technologies are developed with a clear assumption in mind: the waist of the protocol stack is IP and it will be so for the foreseeable future. By fixing the protocol stack waist, engineers can answer a large set of questions, including how to handle conversational traffic (e.g., voice calls) vs. web traffic, how to support multicast, and so on, in a rather straightforward manner. Broadcast, on the other hand, which is inherent in wireless communication is not fully taken advantage of. On the contrary, researchers are often more concerned about introducing mechanisms that ensure that "broadcast storms" do not take down a network. The question of how can broadcast better serve ICN needs has yet to be thoroughly investigated.

Wireless networking is often intertwined with mobility but this is not always the case. In fact, empirical measurements often indicate that many users tend to connect (and remain connected) to a single Wi-Fi access point for considerable amounts of time. A case in point, which is frequently cited in different variations in the ICN literature, is access to a document repository during a meeting. For instance, in a typical IETF working group meeting, a scribe takes notes which are uploaded to a centralized repository (see Figure 1). Subsequently, each meeting participant obtains a copy of the document on their own devices for local use, annotation, and sharing with colleagues that are not present at the meeting. Note that in this example there is no node mobility and that it is not important whether the document with the notes is uploaded in one go at the end

of the session or in a streaming-like fashion as is typical today with online (cloud-based) document processing.

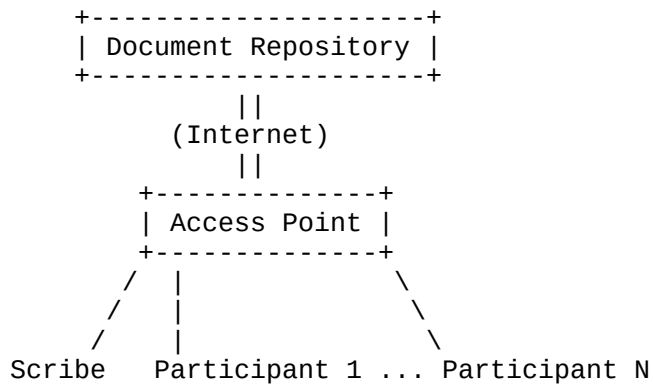


Figure 1: Document sharing during a meeting

In this scenario we observe that the same data object bits (corresponding to the meeting notes) need to traverse the wireless medium at least N+1 times, where N is the number of meeting participants obtaining a copy of the notes. In effect, a broadcast medium is shoehorned into N+1 virtual unicast channels. One could argue that wireless local connectivity is inexpensive, but this is not the critical factor in this example. The actual information exchange wastes N times the available network capacity, no matter what is the spectral efficiency (or the economics) underlying the wireless technology. This waste is a direct result of extending the remote access paradigm from wired to wireless communication, irrespective of the special characteristics of the latter.

It goes without saying that an ICN approach that does not take into consideration the wireless nature of an interface will waste the same amount of resources as a host-centric paradigm. In-network caching at the wireless access point could reduce the amount of data carried over the backhaul link but, if there is no change in the use of the wireless medium, the NDO will still be carried over the wireless ether N+1 times. Intelligent caching strategies, replica placement cooperation and so on simply cannot alleviate this. On the other hand, promiscuous interface operation and opportunistic caching would maximize wireless network capacity utilization in this example.

Arguably, if one designs a future wireless access technology with an information-centric "layer 3" in mind, many of the design choices that are obvious in an all-IP architecture may no longer be valid. Although this is clearly outside the scope of this document, a few

research challenges that the wider community may be interested in include:

- Can we use wireless resources more frugally with the information-centric paradigm than what is possible today in all-IP wireless networks?
- In the context of wireless access, how can we leverage the broadcast nature of the medium in an information-centric network?
- Would a wireless-oriented ICN protocol stack deliver significant performance gains? How different would it be from a wired-oriented ICN protocol stack?
- Is it possible that by changing the network paradigm to ICN we can in practice increase the spectral efficiency (bits/s/Hz) of a wireless network beyond what would be possible with today's host-centric approaches? What would be the impact of doing so with respect to energy consumption?
- Can wireless interface promiscuous operation coupled with opportunistic caching increase ICN performance, and if so, by how much?
- How can a conversational service be supported at least as efficiently as today's state-of-the-art wireless networks deliver?
- What are the benefits from combining ICN with network coding in wireless networks?
- How can MIMO and Coordinated Multipoint Transmission (CoMP) be natively combined with ICN primitives in future cellular networks?

4.6. Rate and Congestion Control

ICN's receiver-driven communication model as described above creates new opportunities for transport protocol design, as it does not rely solely on end-to-end communication from a sender to a requestor. A requested data object can be accessible in multiple different network locations. A node can thus decide how to utilize multiple sources, e.g., by sending parallel requests for the same NDO or by switching sources (or next hops) in a suitable schedule for a series of requests.

In this model, the requestor would control the data rate by regulating its request sending rate and next by performing source/ next-hop selections. Specific challenges depend on the specific ICN approach, but general challenges for receiver-driven transport

protocols (or mechanisms, since dedicated protocols might not be required) include flow and congestion control, fairness, network utilization, stability (of data rates under stable conditions) etc. [HRICP] and [ConTug] describe request rate control protocols and corresponding design challenges.

As mentioned above, the ICN communication paradigm does not depend strictly on end-to-end flows, as contents might be received from in-network caches. The traditional concept of a flow is then somewhat not valid as sub-flows, or flowlets, might be formed on the fly, when fractions of an NDO are transmitted from in-network caches. For a transport layer protocol this is challenging, as any measurement related to this flow as traditionally done by transport protocols such as TCP, can often prove misleading. For example, false Round-Trip Time (RTT) measurements will lead to largely variable average and smoothed RTT values, which in turn will trigger false timeout expirations.

Furthermore, out-of-order delivery is expected to be common in a scenario where parts of a data object are retrieved from in-network caches, rather than from the origin server. Several techniques for dealing with out-of-order delivery have been proposed in the past for TCP, some of which could potentially be modified and re-used in the context of ICN. Further research is needed in this direction though to choose the right technique and adjust it according to the requirements of the ICN architecture and transport protocol in use.

ICN offers routers the possibility to aggregate requests and can use several paths, meaning that there is no such thing as a (dedicated) end-to-end communication path, e.g., a router that receives two requests for the same content at the same time only sends one request to its neighbour. The aggregation of requests has a general impact on transport protocol design and offers new options for employing per-node forwarding strategies and for rethinking in-network resource sharing. [hotnets2014-psaras]

Achieving fairness for requestors can be one challenge as it is not possible to identify the number of requestors behind one particular request. A second problem related to request aggregation is the management of request retransmissions. Generally, it is assumed that a router will not transmit a request if it transmitted an identical request recently and because there is no information about the requestor, the router cannot distinguish the initial request from a client from a retransmission from the same client. In such a situation, how routers can adapt their timers to use the best of the communication paths.

4.7. In-Network Caching

Explicitly named data objects allow for caching at virtually any network element, including routers, proxy caches and end-user devices. In-network caching can therefore improve network performance by fetching content from nodes geographically placed closer to the end-user. Several issues that need further investigation have been identified with respect to in-network caching. In this section we list important challenges that relate to the properties of the new ubiquitous caching system.

4.7.1. Cache Placement

The declining cost of fast memory gives the opportunity to deploy caches in network routers and take advantage of cached NDOs. We identify two approaches to in-network caching, namely, on-path and off-path caching. Both approaches have to consider the issue of cache location. Off-path caching is similar to traditional proxy-caching or CDN server placement. Retrieval of contents from off-path caches requires redirection of requests and, therefore, is closely related to the Request-to-Cache Routing problem discussed later. Off-path caches have to be placed in strategic points within a network in order to reduce the redirection delays and the number of detour hops to retrieve cached contents. Previous research on proxy-caching and CDN deployment is helpful in this case.

On the other hand, on-path caching requires less network intervention and fits more neatly in ICN. However, on-path caching requires line-speed operation, which places more constraints on the design and operation of in-network caching elements. Furthermore, the gain of such a system of on-path in-network caches relies on opportunistic cache hits and has therefore been considered of limited benefit, given the huge amount of contents hosted in the Internet. For this reason, network operators might initially consider only a limited number of network elements to be upgraded to in-network caching elements. The decision on which nodes should be equipped with caches is an open issue and might be based, among others, on topological criteria, or traffic characteristics. These challenges relate to both the Content Placement Problem and the Request-to-Cache Routing Problem discussed below.

In most cases, however, the driver for the implementation, deployment and operation of in-network caches will be its cost. Operating caches at line speed inevitably requires faster memory, which increases the implementation cost. Based on the capital to be invested, ISPs will need to make strategic decisions on the cache placement, which can be driven by several factors, such as: avoid inter-domain/expensive links, centrality of nodes, size of domain and

the corresponding spatial locality of users, traffic patterns in a specific part of the network (e.g., university vs. business vs. fashion district of a city).

4.7.2. Content Placement -- Content-to-Cache Distribution

Given a number of on-path or off-path in-network caching elements, content-to-cache distribution will affect both the dynamics of the system, in terms of request redirections (mainly in case of off-path caches) and the gain of the system in terms of cache hits. A straightforward approach to content placement is on-path placement of contents as they travel from source to destination. This approach reduces the computation and communication overhead of placing content within the network but, on the other hand, might reduce the chances of hitting cached contents. This relates to the Request-to-Cache Routing problem discussed next.

Furthermore, the number of replicas held in the system brings up resource management issues in terms of cache allocation. For example, continuously replicating data objects in all network elements results in redundant copies of the same objects. The issue of redundant replication has been investigated in the past for hierarchical web caches. However, in hierarchical web-caching, overlay systems coordination between the data and the control plane can guarantee increased performance in terms of cache hits. Line-speed, on-path in-network caching poses different requirements and therefore, new techniques need to be investigated. In this direction, reducing the redundancy of cached copies is a study item. However, the issue of coordinated content placement in on-path caches remains open.

The Content-to-Cache Allocation problem relates also to the characteristics of the content to be cached. Popular content might need to be placed where it is going to be requested next. Furthermore, issues of "expected content popularity" or temporal locality need to be taken into account in designing in-network caching algorithms in order for some contents to be given priority (e.g., popular content vs. one-timers). The criteria as to which contents should be given priority in in-network content caches relate also to the business relationships between content providers and network operators. Business model issues will drive some of these decisions on content-to-cache distribution, but such issues are outside the scope of this note and are not discussed here further.

4.7.3. Request-to-Cache Routing

In order to take advantage of cached contents, requests have to be forwarded to the nodes that cache the corresponding contents. This challenge relates to name-based routing, discussed earlier. Requests should ideally follow the path to the cached NDO. However, instructions as to which content is cached where cannot be broadcast throughout the network. Therefore, the knowledge of a NDO location at the time of the request might either not exist, or it might not be accurate (i.e., contents might have been removed by the time a request is redirected to a specific node).

Coordination between the data and the control planes to update information of cached contents has been considered, but in this case scalability issues arise. We therefore, have two options. We either have to rely on opportunistic caching, where requests are forwarded to a server and in case the NDOt is found on the path, then the content is fetched from this node instead of the origin server; or we employ cache-aware routing techniques. Cache-aware routing can either involve both the control and the data plane, or only one of them. Furthermore, cache-aware routing can be done in a domain-wide scale or can involve more than one individual Autonomous System (AS). In the latter case, business relationships between ASes might need to be exploited in order to build a scalable model.

4.7.4. Staleness Detection of Cached NDOs

Due to the largely distributed copies of NDOs in in-network caches, ICN should be able to provide a staleness verification algorithm which provides synchronization of NDOs located at their providers and in-network caching points. Two types of approaches can be considered for this problem, namely direct and indirect approaches.

In the direct approach, each cache looks up certain information in the name of NDO, e.g., timestamp which directly indicates its staleness. This approach is applicable to some NDOs that come from machine-to-machine and Internet of Things scenarios, whose base operation relies on obtaining the latest version of that NDO (i.e., a soil sensor in a farm providing different continuous parameters that are sent to a display or green-house regulation system) [FRESHNESS].

In the indirect approach, each cache consults the publisher of the cached NDO about its staleness before serving it. This approach assumes that the NDO includes the publisher information which can be used to reach the publisher. It is suitable for the NDO whose expiration time is difficult to be set in advance, e.g., a web page which contains main text (that stays the same ever after) and the

interactive sections such as comments or ads (that are updated irregularly).

It is often argued that ignoring stale NDOs in caches and simply providing new names for updated NDOs might be sufficient rather than using a staleness verification algorithm to manage them. However, notifying the new names of updated NDOs to users is not a trivial task. Unless the update is informed to all users at the same time, some users would use the old name although intending to retrieve the updated NDO.

One research challenge is how to design consistency and coherence models for caching NDOs along with their revision handling and updating protocols in a scalable manner.

4.7.5. Cache Sharing by Multiple Applications

When ICN is deployed as a general, application-independent network and cache infrastructure, multiple consumers and producers (representing different applications) would communicate over the same infrastructure. With universal naming schemes or sufficiently unique hash-based identifiers different application could also share identical NDOs in a transparent way.

Depending on the naming, data integrity and data orign authentication approaches, there may be technical and business challenges to share caches across different applications, for example content protection, avoiding cache poisoning, ensuring performance isolation etc. As ICN research matures, these challenges should be addressed more specifically in a dedicated document.

4.8. Network Management

Managing networks has been a core craft in the IP-based host-centric paradigm ever since the technology was introduced in production networks. However, at the onset of IP, management was considered primarily as an add-on. Essential tools that are used daily by networkers, such as ping and traceroute, did not become widely available until more than a decade or so after IP was first introduced. Management protocols, such as SNMP, also became available much later than the original introduction of IP and many still consider them insufficient despite the years of experience we have running host-centric networks. Today, when new networks are deployed network management is considered a key aspect for any operator, a major challenge which is directly reflected in higher operational cost if not done well. If we want ICN to be deployed in infrastructure networks, development of management tools and

mechanisms must go hand-in-hand with the rest of the architecture design.

Although defining an FCAPS (fault, configuration, accounting, performance, security) [ISOIEC-7498-4] management model for ICN is clearly outside the scope of this document, there is a need for creating basic tools early on while ICN is still in the design and experimentation phases that can evolve over time and help network operations centers (NOCs) to define policies, validate that they are indeed used in practice, be notified early on about failures, determine and resolve configuration problems. Authentication, Authorization, Accounting (AAA) as well as performance management, from a NOC perspective, will also need to be considered. Given the expectations for a large number of nodes and unprecedented traffic volumes, automating tasks, or even better employing self-management mechanisms are preferred. The main challenge here is that all tools we have at our disposal today are node-centric, end-to-end oriented, or assuming a packet-stream communication environment. Rethinking reachability and operational availability, for example, can yield significant insights into how information-centric networks will be managed in the future.

With respect to network management we see three different aspects. First, any operator needs to manage all resources available in the network, which can range from node connectivity to network bandwidth availability to in-network storage to multi-access support. In ICN, users will also bring into the network significant resources in terms of network coverage extension, storage, and processing capabilities. Delay Tolerant Networking (DTN) characteristics should also be considered to the degree that this is possible (e.g., content dissemination through data mules). Secondly, given that nodes and links are not at the center of an information-centric network, network management should capitalize on native ICN mechanisms. For example, in-network storage and name resolution can be used for monitoring, while native publish/subscribe functionality can be used for triggering notifications. Finally, management is also at the core of network controlling capabilities by allowing operating actions to be mediated and decided, triggering and activating networking procedures in an optimized way. For example, monitoring aspects can be conjugated with different management actions in a coordinated way, allowing network operations to flow in a concertated manner.

However, the considerations on leveraging intrinsic ICN mechanisms and capabilities to support management operations go beyond a simple mapping exercise. In fact, not only it raises a series of challenges on its own, but also opens up new possibilities for both ICN and "network management" as a concept. For instance, naming mechanisms

are central to ICN intrinsic operations, which are used to identify and reach content under different aspects (e.g., hierarchically structured vs. 'flattish' names). In this way, ICN is decoupled from host-centric aspects on which traditional networking management schemes rely upon. As such, questions are raised which can directly be translated into challenges for network management capability, such as, for example how to address a node or a network segment in a ICN naming paradigm, how to identify which node is connected "where", how to be aware of the node capabilities (i.e., high or low-powered M2M node) and if there is a host-centric protocol running where the management process can also leverage.

But, on the other hand, these same inherent ICN characteristics also allow us to look into network management through a new perspective. By centering its operations around NDOs, one can conceive new management operations addressing, for example, per-content management or access control, as well as analyzing performance per NDO instead of per link or node. Moreover, such considerations can also be used to manage operational aspects of ICN mechanisms themselves. For example, [NDN-MGMT] re-utilizes inherent content-centric capabilities of CCN to manage optimal link connectivity for nodes, in concert with a network optimization process. Conversely, how these content-centric aspects can otherwise influence and impact management in other areas (e.g., security, resilience) is also important, as exemplified in [CCN-ACCESS], where access control mechanisms are integrated into a prototype of the [PURSUIT] architecture.

The set of core research challenges on ICN management include:

- Management and control of NDO reception at the requestor
- Coordination of management information exchange and control between ICN nodes and ICN network control points
- Identification of management and controlling actions and items through information naming
- Relationship between NDOs and host entities identification, i.e., how to identify a particular link, interface, or flow that need to be managed.

4.9. ICN Applications

ICN can be applied to different application domains and is expected to provide benefits for application developers by providing a more suitable interface for application developers (in addition to the other ICN benefits described above). [RFC7476] provides an overview

of relevant application domains at large. This section discusses opportunities and challenges for selected application types.

4.9.1. Web Applications

Intuitively, the ICN request/response communication style seems to be directly mappable to web communication over HTTP. NDO names could be the equivalent of URIs in today's web, proprietary and transparent caching could be obsoleted by ICN in-network caching, and developers could directly use an ICN request/response API to build applications.

Research efforts such as [ICN2014-WEB-NDN] have analysed real-world web applications and ways to implement them in ICN. The most significant insight is that, REST-style web communication heavily relies on transmitting user/application context information in HTTP GET requests, which would have to be mapped to corresponding ICN messages. The challenge in ICN would be how to exactly achieve that mapping. This could be done to some degree by extending name formats or by extending message structure to include cookies and similar context information. The design decisions would need to consider overhead in routers (if larger GET/Interest messages would have to be stored in corresponding tables on routers, for example.

Other challenges include the ability to return different results based on requestor-specific processing in the presence on immutable objects (and name-object bindings) in ICN and the ability for efficient bidirectional communication, which would require some mechanism to name and reach requestor applications.

4.9.2. Video Streaming and Download

One of ICN's prime application areas is video streaming and download where accessing named data, object-level security and in-network storage can fulfil requirements for both video streaming and download. The applicability and benefits of ICN to video has been demonstrated by several prototype developments [ICN2014-AHLGREN-VIDEO-DEMO].

[I-D.irtf-icnrg-videostreaming] discusses the opportunities and challenges for implementing today's' video services such as DASH-based streaming and download over ICN, considering performance requirements, relationship to peer-to-peer live streaming, IPTV and Digital Rights Management (DRM).

In addition to just porting today's video application from a host-centric paradigm to ICN there are also promising opportunities to leverage the ICN network services for redesigning and thus significantly enhancing video access and distribution

[ICNRG-2015-01-WESTPHAL]. For example, ICN store and forward could be leveraged for rate adaptation to achieve maximum throughput and optimal QoE in scenarios with varying link properties, if capacity information is fed back to rate selection algorithms at senders. Other optimizations such as more aggressive prefetching could be performed in the network by leveraging visibility of chunk NDO names and NDO metadata in the network. Moreover, multi-source rate adaptation in combination with network coding could enable better quality of experience, for example in multi-interface/access scenarios where multiple paths from client to upstream caches exist [RFC7476].

4.9.3. Internet of Things

The essence of ICN lies in the name based routing that enables users to retrieve NDOs by the names regardless of their locations. By the definition, ICN is suitable well for IoT applications, where users consume data generated from IoTs without maintaining secure connections to them. The basic put/get style APIs of ICN enable developers to build IoT applications in a simple and fast manner.

On-going efforts such as [I-D.lindgren-icnrg-efficientiot], [I-D.zhang-iot-icn-challenges], [ICN2014-NDNWILD] have addressed the requirements and challenges of ICN for IoT. For instance, many IoT applications depend on a PUSH model where data transmission is initiated by the publisher, and so they can support various real-time applications: emergency alarm, etc. However, ICN does not support the PUSH model in a native manner due to its inherent receiver-driven data transmission mechanism. The challenge would be how to efficiently support the PUSH model in ICN, and so it provides publish/subscribe style APIs for IoT application developers. This could be done by introducing other types of identifiers such as a device identifier or by extending the current request/response communication style, which may result in heavy overhead in ICN routers.

Moreover, key characteristics of the ICN underlying operation also impact important aspects of IoT, such as the caching in content storage of network forwarding entities. This allows the simplification of ICN-based IoT application development, since the network is able to act on named content, generic names provide a way to address content independently of the underlying device (and access) technology, and bandwidth consumption is optimized due to the availability of cached content. However, these aspect raise challenges themselves, concerning the freshness of the information received from the cache in contrast to the last value generated by a sensor, as well as pushing content to specific nodes (e.g., for controlling them), which requires individual addressing for

identification. In addition, due to the heterogeneous nature of IoT nodes, their processing capabilities might not be able to handle the necessary content signing verification procedures.

5. Security Considerations

This document does not impact the security of the Internet. ICN security-related questions related to ICN are discussed in Section 4.2.

6. IANA Considerations

This document presents no IANA considerations.

7. Informative References

[access-control-delegation]
Fotiou, N., Marias, G., and G. Polyzos, "Access control enforcement delegation for information-centric networking architectures", Proceedings of the second edition of the ICN workshop on Information-centric networking (ICN '12) Helsinki, Finnland, 2012.

[BACKSCATTER]
Waehlisch, M., Schmidt, TC., and M. Vahlenkamp, "Backscatter from the Data Plane - Threats to Stability and Security in Information-Centric Network Infrastructure", Computer Networks Vol 57, No. 16, pp. 3192-3206, November 2013.

[BREADCRUMBS]
Rosensweig, E. and J. Kurose, "Breadcrumbs: Efficient, Best-Effort Content Location in Cache Networks", In Proceedings of the IEEE INFOCOM 2009, April 2009.

[CCN] Jacobson, , K, , D, , F, , H, , and L, "Networking Named Content", CoNEXT 2009 , December 2009.

[CCN-ACCESS]
Fotiou, N., Marias, G., and G. Polyzos, "Access control enforcement delegation for information-centric networking architectures", In Proceedings of the second edition of the ICN workshop on Information-centric networking (ICN '12). ACM, New York, NY, USA, 85-90., 2012.

[Chaum] Chaum, D. and E. van Heijst, "Group signatures", In Proceedings of EUROCRYPT, 1991.

[COMPACT] Cowen, L., "Compact routing with minimum stretch", In Journal of Algorithms, vol. 38, pp. 170--183, 2001.

[ConTug] Arianfar, S., Nikander, P., Eggert, L., Ott, J., and W. Wong, "ConTug: A Receiver-Driven Transport Protocol for Content-Centric Networks", Technical Report Aalto University Comnet, 2011.

[DONA] Koponen, T., Ermolinskiy, A., Chawla, M., Kim, K., gon Chun, B., and S. Shenker, "A Data-Oriented (and Beyond) Network Architecture", In Proceedings of SIGCOMM 2007, August 2007.

[encryption-ac] Kurihara, J., Uzun, E., and C. Wood, "An Encryption-Based Access Control Framework for Content-Centric Networking", IFIP Networking 2015 Toulouse, France, 2015, September 2015.

[FRESHNESS] Quevedo, J., Corujo, D., and R. Aguiar, "Consumer Driven Information Freshness Approach for Content Centric Networking", IEEE INFOCOM Workshop on Name-Oriented Mobility Toronto, Canada, 2014, May 2014.

[GREEDY] Papadopoulos, F., Krioukov, D., Boguna, M., and A. Vahdat, "Greedy forwarding in dynamic scale-free networks embedded in hyperbolic metric spaces", In Proceedings of the IEEE INFOCOM, San Diego, USA, 2010.

[hotnets2014-psaras] Psaras, I., Saino, L., and G. Pavlou, "Revisiting Resource Pooling: The case of In-Network Resource Sharing", ACM HotNets Los Angeles, USA, 2014, October 2014.

[HRICP] Carofiglio, G., Gallo, M., and L. Muscariello, "Joint hop-by-hop and receiver-driven interest control protocol for content-centric networks", In Proceedings of ACM SIGCOMM ICN 2012, DOI 10.1145/2342488.2342497, 2012.

[I-D.irtf-icnrg-videostreaming] Lederer, S., cedric.westphal@huawei.com, c., Mueller, C., Detti, A., Corujo, D., Wang, J., Montpetit, M., Murray, N., aytav.azgin, a., LIU, S., Timmerer, C., and D. Posch, "Adaptive Video Streaming over ICN", draft-irtf-icnrg-videostreaming-05 (work in progress), December 2015.

[I-D.lindgren-icnrg-efficientiot]
Lindgren, A., Abdesslem, F., Ahlgren, B., Schelen, O., and A. Malik, "Applicability and Tradeoffs of Information-Centric Networking for Efficient IoT", draft-lindgren-icnrg-efficientiot-03 (work in progress), July 2015.

[I-D.zhang-iot-icn-challenges]
Zhang, Y., Raychadhuri, D., Grieco, L., Baccelli, E., Burke, J., Ravindran, R., and G. Wang, "ICN based Architecture for IoT - Requirements and Challenges", draft-zhang-iot-icn-challenges-02 (work in progress), August 2015.

[ICN2014-AHLGREN-VIDEO-DEMO]
Ahlgren, B., Jonasson, A., and B. Ohlman, "Demo Overview: HTTP Live Streaming over NetInf Transport", ACM SIGCOMM Information-Centric Networking Conference Paris, France, 2014, September 2014.

[ICN2014-NDNWILD]
Baccelli, E., Mehlis, C., Hahm, O., Schmidt, T., and M. Waehlisch, "Information Centric Networking in the IoT: Experiments with NDN in the Wild", ACM SIGCOMM Information-Centric Networking Conference Paris, France, 2014, September 2014.

[ICN2014-WEB-NDN]
Moiseenko, I., Stapp, M., and D. Oran, "Communication Patterns for Web Interaction in Named Data Networking", ACM SIGCOMM Information-Centric Networking Conference Paris, France, 2014, September 2014.

[ICNNAMING]
Ghodsi, A., Koponen, T., Rajahalme, J., Sarolahti, P., and S. Shenker, "Naming in Content-Oriented Architectures", In Proceedings ACM SIGCOMM Workshop on Information-Centric Networking (ICN), 2011.

[ICNRG-2015-01-WESTPHAL]
Westphal, C., "Video over ICN", IRTF ICNRG Meeting Cambridge, Massachusetts, USA, 2015, URI http://www.ietf.org/proceedings/interim/2015/01/13/icnrg/slides/slides-interim-2015-icnrg-1-0.pptx, January 2015.

[ICNSURVEY]
Ahlgren, B., Dannewitz, C., Imbrenda, C., Kutscher, D., and B. Ohlman, "A Survey of Information-Centric Networking", In Communications Magazine, IEEE , vol.50, no.7, pp.26-36, DOI 10.1109/MCOM.2012.6231276, 2012.

[ISOIEC-7498-4]
ISO, , "Information Processing Systems -- Open Systems Interconnection -- Basic Reference Model -- Part 4: Management Framework", URI http://standards.iso.org/ittf/PubliclyAvailableStandards/s014258_ISO_IEC_7498-4_1989(E).zip, November 1989.

[MANI] Pentikousis, K. and T. Rautio, "A multiaccess Network of Information", WoWMoM 2010, IEEE , June 2010.

[MDHT] D'Ambrosio, M., Dannewitz, C., Karl, H., and V. Vercellone, "MDHT: A hierarchical name resolution service for information-centric networks", ACM SIGCOMM workshop on Information-centric networking Toronto, Canada, 2011, August 2011.

[MMIN] Pentikousis, K. and P. Bertin, "Mobility management in infrastructure networks", Internet Computing, IEEE, vol. 17, no. 5, pp. 74-79 , October 2013.

[ndn-controlled-sharing]
Yu, Y., "Controlled Sharing of Sensitive Content", IRTF ICNRG Meeting San Francisco, USA, 2015, URI https://www.ietf.org/proceedings/interim/2015/10/03/icnrg/slides/slides-interim-2015-icnrg-4-8.pdf, October 2015.

[NDN-MGMT]
Corujo, D., Aguiar, R., Vidal, I., and J. Garcia-Reinoso, "A named data networking flexible framework for management communications", Communications Magazine, IEEE , vol.50, no.12, pp.36-43 , December 2012.

[PURSUIT] Fotiou et al., N., "Developing Information Networking Further: From PSIRP to PURSUIT", In Proceedings of Proc. BROADNETS. ICST, 2010.

[RANDOM] Gkantsidis, C., Mihail, M., and A. Saberi, "Random walks in peer-to-peer networks: algorithms and evaluation", In Perform. Eval., vol. 63, pp. 241--263, 2006.

[RFC2002] Perkins, C., Ed., "IP Mobility Support", RFC 2002, DOI 10.17487/RFC2002, October 1996, <http://www.rfc-editor.org/info/rfc2002>.

[RFC4838] Cerf, V., Burleigh, S., Hooke, A., Torgerson, L., Durst, R., Scott, K., Fall, K., and H. Weiss, "Delay-Tolerant Networking Architecture", RFC 4838, DOI 10.17487/RFC4838, April 2007, <http://www.rfc-editor.org/info/rfc4838>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <http://www.rfc-editor.org/info/rfc5246>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <http://www.rfc-editor.org/info/rfc5280>.

[RFC5944] Perkins, C., Ed., "IP Mobility Support for IPv4, Revised", RFC 5944, DOI 10.17487/RFC5944, November 2010, <http://www.rfc-editor.org/info/rfc5944>.

[RFC6920] Farrell, S., Kutscher, D., Dannewitz, C., Ohlman, B., Keranen, A., and P. Hallam-Baker, "Naming Things with Hashes", RFC 6920, DOI 10.17487/RFC6920, April 2013, <http://www.rfc-editor.org/info/rfc6920>.

[RFC7476] Pentikousis, K., Ed., Ohlman, B., Corujo, D., Boggia, G., Tyson, G., Davies, E., Molinaro, A., and S. Eum, "Information-Centric Networking: Baseline Scenarios", RFC 7476, DOI 10.17487/RFC7476, March 2015, <http://www.rfc-editor.org/info/rfc7476>.

[RFC7696] Housley, R., "Guidelines for Cryptographic Algorithm Agility and Selecting Mandatory-to-Implement Algorithms", BCP 201, RFC 7696, DOI 10.17487/RFC7696, November 2015, <http://www.rfc-editor.org/info/rfc7696>.

[SEEN] Pentikousis, K., "In search of energy-efficient mobile networking", Communications Magazine, IEEE, vol. 48, no. 1, pp. 95-103 , January 2010.

Appendix A. Acknowledgments

The authors would like to thank Georgios Karagiannis for providing suggestions on QoS research challenges, Dimitri Papadimitriou for feedback on the routing section, and Joerg Ott and Stephen Farrell for reviewing the whole document.

Authors' Addresses

Dirk Kutscher (editor)
NEC
Kurfuersten-Anlage 36
Heidelberg
Germany

Email: kutscher@neclab.eu

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

Phone: +81-42-327-6582
Email: suyong@nict.go.jp

Kostas Pentikousis
EICT GmbH
Torgauer Strasse 12-15
Berlin 10829
Germany

Email: k.pentikousis@eict.de

Ioannis Psaras
University College London, Dept. of E.E. Eng.
Torrington Place
London WC1E 7JE
United Kingdom

Email: i.psaras@ucl.ac.uk

Daniel Corujo
Universidade de Aveiro
Instituto de Telecomunicacoes, Campus Universitario de Santiago
Aveiro P-3810-193
Portugal

Email: dcorujo@av.it.pt

Damien Saucez
INRIA
2004 route des Lucioles - BP 93
Sophia Antipolis 06902 Cedex
France

Email: damien.saucez@inria.fr

Thomas C. Schmidt
HAW HAmburg
Berliner Tor 7
Hamburg 20099
Germany

Email: t.schmidt@ieee.org

Matthias Waehlisch
FU Berlin
Takustr. 9
Berlin 14195
Germany

Email: waehlisch@ieee.org

ICNRG
Internet-Draft
Intended Status: Informational
Expires: October 14, 2016

K. Pentikousis, Ed.
EICT
B. Ohlman
Ericsson
E. Davies
Trinity College Dublin
S. Spirou
Intracom Telecom
G. Boggia
Politecnico di Bari

April 12, 2016

Information-centric Networking: Evaluation and Security Considerations
draft-irtf-icnrg-evaluation-methodology-05

Abstract

This document presents a number of considerations regarding evaluating Information-centric Networking (ICN) and sheds some light on the impact of ICN on network security. It also surveys the evaluation tools currently available to researchers in the ICN area and provides suggestions regarding methodology and metrics.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

Table of Contents

1. Introduction

Information-centric Networking (ICN) is a networking concept that arose from the desire to align the operation model of a network with the model of its typical use. For TCP/IP networks, this implies changing the mechanisms of data access and transport from a host-to-host model to a user-to-information model. The premise is that the effort invested in changing models will be offset, or even surpassed, by the potential of a "better" network. However, such a claim can be validated only if it is quantified.

Different ICN approaches are evaluated in the peer-reviewed literature using a mixture of theoretical analysis, simulation and emulation techniques, and empirical (testbed) measurements. The specific methodology employed may depend on the experimentation goal, e.g., whether one wants to evaluate scalability, quantify resource utilization, or analyze economic incentives. In addition, though, we observe that ease and convenience of setting up and running experiments can sometimes be a factor in published evaluations. As discussed in [RFC7476], the development phase that ICN is going through and the plethora of approaches to tackle the hardest problems make this a very active and growing research area but, on the downside, it also makes it more difficult to compare different proposals on an equal footing.

Performance evaluation using actual network deployments has the advantage of realistic workloads and reflects the environment where the service or protocol are to be deployed. In the case of ICN, however, it is not currently clear what qualifies as a "realistic workload". Trace-based analysis of ICN is in its infancy, and more work is needed towards defining characteristic workloads for ICN evaluation studies. Accordingly, the experimental process and the evaluation methodology per se are actively being researched for different ICN architectures. Numerous factors affect the experimental results, including the topology selected, the background traffic that an application is being subjected to, network conditions such as available link capacities, link delays, and loss-rate characteristics throughout the selected topology; failure and disruption patterns; node mobility and the diversity of devices used.

The goal of this document is to summarize evaluation guidelines and tools alongside suggested data sets and high-level approaches. We expect this to be of interest to the ICN community as a whole as it can assist researchers and practitioners alike to compare and contrast different ICN designs against each other, as well as against the state of the art in host-centric solutions, and identify the respective strengths and weaknesses. We note that apart from the technical evaluation of the functionality of an ICN architecture, its future success will be largely driven by its deployability and economic viability. Therefore, ICN evaluations should assess

incremental deployability in the existing network environment together with a view of how the technical functions will incentivize deployers to invest in the capabilities that allow the architecture to spread across the network.

This document has been produced by the IRTF Information-centric Networking Research Group (ICNRG). The main objective of the ICNRG is to couple ongoing ICN research in the above areas with solutions that are relevant for evolving the Internet at large. The ICNRG produce documents that provides guidelines for experimental activities in the area of ICN so that different, alternative solutions can be compared consistently, and information sharing accomplished for experimental deployments. This document incorporates input from ICNRG participants and their corresponding text contributions; it has been reviewed by several ICNRG active participants (see section 7), and represents the consensus of the research group. That said, note that this document does not constitute an IETF standard; see also [RFC5743].

The remainder of this document is organized as follows. Section 2 presents various techniques and considerations for evaluating different ICN architectures. Section 3 discusses the impact of ICN on network security. Section 4 surveys the tools currently available to ICN researchers.

2. Evaluation Considerations

It is clear that the way we evaluate IP networks will not be directly applicable for evaluating ICN. In IP the focus is on the performance and characteristics of end-to-end connections between a source and a destination. In ICN the "source" responding to a request can be any ICN node in the network and may change from request to request. This makes it difficult to use concepts like delay and throughput in a traditional way. In addition evaluating resource usage in ICN is a more complicated task as memory used for caching affects delays and use of transmission resources, see the discussion on resource equivalents at the end of this section.

There are two major types of evaluations of ICN that we see a need to make. One type is to compare ICN to traditional networking and one type is to compare different ICN implementations and approaches against each other.

In this section we details some of the functional components needed when evaluating different ICN implementations and approaches.

2.1. Topology Selection

There’s a wealth of earlier work on topology selection for simulation and performance evaluation of host-centric networks. While the classic dumbbell topology is regarded as inappropriate for ICN, most ICN studies so far have been based on that earlier work for host-centric networks [RFC7476]. However, there is no single topology that can be used to easily evaluate all aspects of ICN. Therefore, one should choose from a range of topologies depending on the focus of the evaluation.

For scalability and resilience studies, there is a wide range of synthetic topologies, such as the Barabasi-Albert model [BA] and the Watts-Strogatz small-world topology [WATTS]. These allow experiments to be performed whilst controlling various key parameters (e.g., node degree). These synthetic topologies are appropriate in the general case, as there are no practical assurances that a future information-centric network will share the same topology with today’s networks.

When studies look at cost (e.g., transit cost) or migration to ICN, realistic topologies should be used. These can be inferred from Internet traces, such as the CAIDA Macroscopic Internet Topology Data Kit (http://www.caida.org/data/active/internet-topology-data-kit) and Rocketfuel (http://www.cs.washington.edu/research/networking/rocketfuel). A problem is the large size of the topology (approx. 45K ASes, close to 200K links), which may limit the scalability of the employed evaluation tool. Katsaros et al. [ICNScale] address this problem by using scaled down topologies created following the methodology described in [COMPLEX].

Studies that focus on node or content mobility can benefit from topologies and their dynamic aspects as used in the DTN community. As mentioned in [RFC7476], DTN traces are available to be used in such ICN evaluations.

As with host-centric topologies, defining just a node graph will not be enough for most ICN studies. The experimenter should also clearly define and list the respective matrices that correspond to the network, storage and computation capacities available at each node as well as the delay characteristics of each link [Montage]. Real values for such parameters can be taken from existing platforms such as iPlane (http://iplane.cs.washington.edu). Synthetic values could be produced with specific tools [DELAY].

2.2. Traffic Load

In this subsection we provide a set of common guidelines, in the form of what we will refer to as a content catalog for different scenarios. This catalog, which is based on previously published

work, could be used to evaluate different ICN proposals, for instance, on routing, congestion control, and performance, and can be considered as other kinds of ICN contributions emerge. As we are still lacking ICN-specific traffic workloads we can currently only extrapolate from today's workloads. A significant challenge then relates to the identification of the applications contributing to the observed traffic (e.g., Web or peer-to-peer), as well as to the exact amount of traffic they contribute to the overall traffic mixture. Efforts in this direction can take heed from today's traffic mix comprising web, peer-to-peer file sharing, and User Generated Content (UGC) platforms (e.g., YouTube), as well as Video on Demand (VoD) services. Publicly available traces for these include those available from web sites such as http://multiprobe.ewi.tudelft.nl/multiprobe.html, http://an.kaist.ac.kr/traces/IMC2007.html, and http://traces.cs.umass.edu/index.php/Network/Network.

Taking a more systematic approach, and with the purpose of modeling the traffic load, we can resort to measurement studies that investigate the composition of Internet traffic, such as [1][2]. In [1] a large scale measurement study was performed, with the purpose of studying the traffic crossing inter-domain links. The results indicate the dominance of Web traffic, amounting to 52% over all measured traffic. However, Deep Packet Inspection (DPI) techniques reveal that 25-40% of all HTTP traffic actually carries video traffic. Results from DPI techniques also reveal the difficulty in correctly identifying the application type in the case of P2P traffic: mapping observed port numbers to well-known applications shows P2P traffic constituting only 0.85% of overall traffic, while DPI raises this percentage to 18.32% [1]. Relevant studies on a large ISP show the percentage of P2P traffic ranging from 17 to 19% of overall traffic [2]. Table I provides an overview of these figures. The "other" traffic type denotes traffic that cannot be classified in any of the first three application categories, and consists of unclassified traffic and traffic heavily fragmented into several applications (e.g., 0.17% DNS traffic).

Table I. Traffic type ratios of total traffic [1, 2]

Traffic Type	Ratio
Web	31-39%
P2P	17-19%
Video	13-21%
Other	29-31%

The content catalog for each type of traffic can be characterized by a specific set of parameters:

a) The cardinality of the estimated content catalog.

b) The size of the exchanged contents (either chunks or entire named information objects).

c) The popularity of objects expressed in their request frequency.

In most application types, the popularity distribution follows some power law, indicating that a small number of information items trigger a large proportion of the entire set of requests. The exact shape of the power law popularity distribution directly impacts the performance of the underlying protocols. For instance, highly skewed popularity distributions (e.g., a Zipf-like distribution with a high slope value) favor the deployment of caching schemes, since caching a very small set of information items can dramatically increase the cache hit ratio.

Several studies in the past few years have stated that Zipf’s law is the discrete distribution that best represents the request frequency in a number of application scenarios, ranging from the Web to video on demand (VoD) services. The key aspect of this distribution is that the frequency of a content request is inversely proportional to the rank of the content itself, i.e., the smaller the rank, the higher the request frequency. If we denote with M the content catalog cardinality and with 1 <= i <= M the rank of the i-th most popular content, we can express the probability of requesting the content with rank "i" as:

$$P(X=i) = \left(1/i^{(alpha)} \right) / C, \text{ with } C = SUM(1 / j^{(alpha)}), alpha > 0$$

where the sum is obtained considering all values of j, 1 <= j <= M.

A recent analysis of HTTP traffic showed that content popularity is better reflected by a trimodal distribution model in which the head and tail of a Zipf distribution (with slope value 0.84) are replaced by two discrete Weibull distributions with shape parameter values 0.5 and 0.24 respectively [IMB2014].

A variation of the Zipf distribution, termed the Mandelbrot-Zipf distribution was suggested [P2PMod] to better model environments where nodes can locally store previously requested content. For example, it was observed that peer-to-peer file sharing applications typically exhibited a 'fetch-at-most-once' style of behavior. This is because peers tend to persistently store the files they download, a behavior that may also be prevalent in ICN.

Popularity can also be characterized in terms of:

a) The temporal dynamics of popularity, i.e., how requests are distributed in time. The popularity distribution expresses the number of requests submitted for each information item participating into a certain workload. However, they do not describe how these requests are distributed in time. This aspect is of primary importance when considering the performance of caching schemes since the ordering of the requests obviously affects the contents of a cache. For example, with a Least Frequently Used (LFU) cache replacement policy, if all requests for a certain item are submitted close in time, the item is unlikely to be evicted from the cache, even by a (globally) more popular item whose requests are more evenly distributed in time. The temporal ordering of requests gains even more importance when considering workloads consisting of various applications, all competing for the same cache space.

b) The spatial locality of popularity i.e., how requests are distributed throughout a network. The importance of spatial locality relates to the ability to avoid redundant traffic in the network. If requests are highly localized in some area of the entire network, then similar requests can be more efficiently served with mechanisms such as caching and/or multicast i.e., the concentration of similar requests in a limited area of the network allows increasing the perceived cache hit ratios at caches in the area and/or the traffic savings from the use of multicast. Table II provides an overview of distributions that can be used to model each of the identified traffic types i.e., Web, Video (based on YouTube measurements) and P2P (based on BitTorrent measurements). These distributions are the outcome of a series of modeling efforts based on measurements of real traffic workloads

[3][4][5][6][7][8][9][10][11][12][13]. A tool for the creation of synthetic workloads following these models, and also allowing the generation of different traffic mixes is described in [14].

Table II. Overview of traffic types models

	Object Size	Temporal Locality	Popularity Distribution
Web	Concatenation of Lognormal (body) and Pareto (tail) [7,8]	Ordering via LRU stack model [5] Exact timing via exponential distribution [6]	Zipf: p(i)=K/i^a i: popularity rank N: total items K: 1/Sum(1/i^a) a: distribution slope values 0.64-0.84 [3][4]
VoD	Duration/size: Concatenated normal, most videos ~330 kb/s [13]	No analytical models Random distribution across total duration	Weibull: k=0.513, lambda=6010 Gamma: k=0.372, theta=23910 [12]
P2P	Wide variation on torrent sizes [9]. No analytical models exist: Sample a real BitTorrent [11] distribution or use fixed value	Mean arrival rate of 0.9454 torrents/hour Peers in a swarm arrive as l(t)= l0*e^(-t/tau) l0: initial arrival rate (87.74 average) tau: object popularity (1.16 average)* [10]	Mandelbrot-Zipf [9]: p(i)=K/((i+q)/a) q: plateau factor, 5 to 100. Flatter head than in Zipf-like distribution (where q=0)

* Random ordering of swarm births (first request). For each swarm calculate a different tau. Based on average tau and object popularity. Exponential decay rule for subsequent requests.

Table III summarizes the content catalog. With this shared point of reference, the use of the same set of parameters (depending on the scenario of interest) among researchers will be eased, and different proposals could be compared on a common base.

Table III. Content catalog

Traffic Load	Catalog Size [L1][L2][L3][L5]	Mean Object Size [L4][L5][L7][L8][L9][L10]	Popularity Distribution [L3][L5][L6][L11][L12]
Web	10^12	Chunk: 1-10 kB	Zipf with 0.64 <= alpha <= 0.83
File sharing	5x10^6	Chunk: 250-4096 kB Object: ~800 MB	Zipf with 0.75 <= alpha<= 0.82
UGC	10^8	Object: ~10 MB	Zipf, alpha >= 2
VoD (+HLS) (+DASH)	10^4	Object: ~100 MB ~1 kB (*) ~5.6 kB (*)	Zipf, 0.65 <= alpha <= 1

UGC = User Generated Content VoD = Video on Demand

(*) Using adaptive video streaming (e.g., HLS and DASH), with an optimal segment length (2 s for DASh and 10 s for HLS) and a bitrate of 4500 kbps [W16][Led12]

2.3. Choosing Relevant Metrics

Quantification of network performance requires a set of standard metrics. These metrics should be broad enough so they can be applied equally to host-centric and information-centric (or other) networks. This will allow reasoning about a certain ICN approach in relation to an earlier version of the same approach, to another ICN approach or to the incumbent host-centric approach. It will therefore be less difficult to gauge optimization and research direction. On the other hand, the metrics should be targeted to network performance only and should avoid unnecessary expansion into the physical and application layers. Similarly, at this point, it is more important to capture as metrics only the main figures of merit and to leave more esoteric and less frequent cases for the future.

To arrive at a set of relevant metrics, it would be beneficial to look at the metrics used in existing ICN approaches, such as CCN [CCN] [VoCCN] [NDNP], NetInf [4WARD6.1] [4WARD6.3] [SAIL-B2] [SAIL-B3], PURSUIT [PRST4.5], COMET [CMT-D5.2] [CMT-D6.2], Connect [SHARE] [RealCCN], and CONVERGENCE [ICN-Web] [ICN-Scal] [ICN-Tran]. The

metrics used in these approaches fall into two categories: metrics for the approach as a whole, and metrics for individual components (name resolution, routing, and so on). Metrics for the entire approach are further subdivided into traffic and system metrics. It is important to note that the various approaches do not name or define metrics consistently. This is a major problem when trying to find metrics that allow comparison between approaches. For the purposes of exposition, we have tried to smooth over differences by classifying similarly defined metrics under the same name. Also, due to space constraints, we have chosen to report here only the most common metrics between approaches. For more details the reader should consult the references for each approach.

Traffic metrics in existing ICN approaches are summarized in Table IV. These are metrics for evaluating an approach mainly from the perspective of the end user, i.e., the consumer, provider, or owner of the content or service. Depending on the level where these metrics are measured, we have made the distinction into user, application and network-level traffic metrics. So for example, network-level metrics are mostly focused on packet characteristics, whereas user-level metrics can cover elements of human perception. The approaches do not make this distinction explicitly, but we can see from the table that CCN and NetInf have used metrics from all levels, PURSUIT and COMET have focused on lower-level metrics, and Connect and CONVERGENCE opted for higher-level metrics. Throughput and download time seem to be the most popular metrics altogether.

Table IV. Traffic metrics used in ICN evaluations

	User	Application		Network	
	Download time	Goodput	Startup latency	Throughput	Packet delay
CCN	x	x		x	x
NetInf	x		x	x	x
PURSUIT			x	x	x
COMET			x	x	
Connect	x				
CONVERGENCE	x	x			

While traffic metrics are more important for the end user, the owner or operator of the networking infrastructure is normally more interested in system metrics, which can reveal the efficiency of an approach. The most common system metrics used are: protocol overhead, total traffic, transit traffic, cost savings, router cost, and router energy consumption.

Besides the traffic and systems metrics that aim to evaluate an approach as a whole, all surveyed approaches also evaluate the performance of individual components. Name resolution, request/data routing, and data caching are the most typical components, as summarized in Table V. FIB size and path length, i.e., the routing component metrics, are almost ubiquitous among approaches, perhaps due to the networking background of the involved researchers. That might be also the reason for the sometimes decreased focus on traffic and system metrics, in favor of component metrics. It can certainly be argued that traffic and system metrics are affected by component metrics, however no approach has made the relationship clear. With this in mind and taking into account that traffic and system metrics are readily useful to end users and network operators, we will restrict ourselves to those in the following sections.

Table V. Component metrics in existing ICN approaches

	Resolution		Routing		Cache	
	Resolution time	Request rate	FIB size	Path length	Size	Hit ratio
CCN	x		x	x	x	x
NetInf	x	x		x		x
PURSUIT			x	x		
COMET	x	x	x	x		x
CONVERGENCE		x	x		x	

Before proceeding, we should note that we would like our metrics to be applicable to host-centric networks as well. Standard metrics already exist for IP networks and it would certainly be beneficial to take them into account. It is encouraging that many of the metrics used by existing ICN approaches can also be used on IP networks and that all of the approaches have tried on occasion to draw the parallels.

2.3.1. Traffic Metrics

The IETF has been working for more than a decade on devising metrics and methods for measuring the performance of IP networks. The work has been carried out largely within the IP performance metrics (IPPM) working group, guided by a relevant framework [RFC2330]. IPPM metrics include delay, delay variation, loss, reordering, and duplication. While the IPPM work is certainly based on packet-switched IP networks, it is conceivable that it can be modified and extended to cover ICN networks as well. However, more study is necessary to turn this claim into a certainty. Many experts have toiled for a long time on devising and refining the IPPM metrics and methods, so it would be an advantage to use them for measuring ICN performance. In addition, said metrics and methods work already for host-centric networks, so comparison with information-centric networks would entail only the ICN extension of the IPPM framework. Finally, an important benefit of measuring the transport performance of a network at its output, using QoS metrics such as IPPM, is that it can be done mostly without any dependence to applications.

Another option for measuring transport performance would be to use Quality of Service (QoS) metrics, not at the output of the network like with IPPM, but at the input to the application. For live video streaming application the relevant metrics would be startup latency, playout lag and playout continuity. The benefit of this approach is that it abstracts away all details of the underlying transport network, so it can be readily applied to compare between networks of different concepts (host-centric, information-centric, or other). As implied earlier, the drawback of the approach is its dependence on the application, so it is likely that different types of applications will require different metrics. It might be possible to identify standard metrics for each type of application, but the situation is not as clear as with IPPM metrics and further investigation is necessary.

At a higher level of abstraction, we could measure the network’s transport performance at the application output. This entails measuring the quality of the transported and reconstructed information as perceived by the user during consumption. In such an instance we would use Quality of Experience (QoE) metrics, which are by definition dependent on the application. For example, the standardized methods for obtaining a Mean Opinion Score (MOS) for VoIP (e.g., ITU-T P.800) is quite different from those for IPTV (e.g., PEVQ). These methods are notoriously hard to implement, as they involve real users in a controlled environment. Such constraints can be relaxed or dropped by using methods that model human perception under certain environments, but these methods are typically intrusive. The most important drawback of measuring

network performance at the output of the application is that only one part of each measurement is related to network performance. The rest is related to application performance, e.g., video coding, or even device capabilities, both of which are irrelevant to our purposes here and are generally hard to separate. We therefore see the use of QoE metrics in measuring ICN performance as a poor choice at this stage.

2.3.2. System Metrics

Overall system metrics that need to be considered include reliability, scalability, energy efficiency, and delay/disconnection tolerance. In deployments where ICN is addressing specific scenarios, relevant system metrics could be derived from current experience. For example, in IoT scenarios, which were discussed earlier in [RFC7476], it is reasonable to consider the current generation of sensor nodes, sources of information, and even measurement gateways (e.g., for smart metering at homes) or smartphones. In this case, ICN operation ought to be evaluated with respect not only to overall scalability and network efficiency, but also the impact on the nodes themselves. Karnouskos et al. [SensReqs] provide a comprehensive set of sensor and IoT-related requirements, for example, which include aspects such as resource utilization, service life-cycle management and device management.

Additionally, various specific metrics are also critical in constrained environments, such as processing requirements, signaling overhead, and memory allocation for caching procedures in addition to power consumption and battery lifetime. For gateways, which typically act as a point of service to a large number of nodes and have to satisfy the information requests from remote entities we need to consider scalability-related metrics, such as frequency and processing of successfully satisfied information requests.

Finally, given the in-network caching functionality of ICNs, efficiency and performance metrics of in-network caching have to be defined. Such metrics will need to guide researchers and operators regarding the performance of in-network caching algorithms. A first step on this direction has been made in [L9]. The paper proposes a formula that approximates the proportion of time that a content stays in a network cache. The model takes as input the rate of requests for a given content (the Content of Interest) and the rate of requests for all other contents that go through the given network element (router) and move the CoI down in the (LRU) cache. The formula takes also into account the size of the cache of this router.

The output of the model essentially reflects the probability that the

CoI will be found in a given cache. An initial study [L9] is applied to the CCN/NDN framework, where contents get cached at every node they traverse. The formula according to which the probability or proportion is calculated is given by:

pi = [mu/(mu+lambda)]^N,

where lambda is the request rate for CoI, mu is the request rate for contents that move CoI down the cache and N is the size of the cache (in slots).

The formula can be used to assess the caching performance of the system and can also potentially be used to identify the gain of the system due to caching. This can then be used to compare against gains by other factors, e.g., addition of extra bandwidth in the network.

2.4. Resource Equivalence and Tradeoffs

As we have seen above, every ICN network is built from a set of resources, which include link capacities, different types of memory structures and repositories used for storing named data objects and chunks temporarily (i.e., caching) or persistently, as well as name resolution and other lookup services. Complexity and processing needs in terms of forwarding decisions, management (e.g., need for manual configuration, explicit garbage collection, and so on), and routing (i.e., amount of state needed, need for manual configuration of routing tables, support for mobility, etc.) set the stage for a range of engineering tradeoffs.

In order to be able to compare different ICN approaches it would be beneficial to be able to define equivalence in terms of different resources which today are considered incomparable. For example, would provisioning an additional 5 Mb/s link capacity lead to better performance than adding 100 GB of in-network storage? Within this context one would consider resource equivalence (and the associated tradeoffs) for example for cache hit ratios per GB of cache, forwarding decision times, CPU cycles per forwarding decision, and so on.

3. ICN Security Aspects

The introduction of an information-centric networking architecture and the corresponding communication paradigm results in changes to many aspects of network security. These will affect all scenarios described in [RFC7476]. Additional evaluation will be required to ensure relevant security requirements are appropriately met by the

implementation of the chosen architecture in the various scenarios.

The ICN security aspects described in this document is complementing the ICN security challenges outlined in the "ICN Research Challenges" document [ICNCHA]. [***EDITORS NOTE: The referenced document is expected to have become an RFC by the time this document is published, this reference should thus be updated before publication of this document.***]

The ICN architectures currently proposed have concentrated on authentication of delivered content to ensure the integrity of the content. However the approaches are primarily applicable to freely accessible content that does not require access authorization, although they will generally support delivery of encrypted content.

The introduction of widespread caching mechanisms may also provide additional attack surfaces. The caching architecture to be used also needs to be evaluated to ensure that it meets the requirements of the usage scenarios.

In practice, the work on security in the various ICN research projects has been heavily concentrated on authentication of content. Work on authorization, access control, privacy and security threats due to the expanded role of in-network caches has been quite limited. For Example, a roadmap for improving the security model in NetInf can be found in [NETINFSC]. As secure communications on the Internet are becoming the norm, major gaps in ICN security aspects are bound to undermine the adoption of ICN.

In the rest of this section we briefly consider the issues and provide pointers to the work that has been done on the security aspects of the architectures proposed.

3.1. Authentication

For fully secure content distribution, content access requires that the receiver needs to be able to reliably assess:

validity: is it a complete, uncorrupted copy of what was originally published;

provenance: can the receiver identify the publisher, and, if so, whether it and the source of any cached version of the document can be adequately trusted; and

relevance: is the content an answer to the question that the receiver asked.

All ICN architectures considered in this document primarily target the validity requirement using strong cryptographic means to tie the content request name to the content. Provenance and relevance are directly targeted to varying extents: There is a tussle or trade-off between simplicity and efficiency of access and level of assurance of all these traits. For example, maintaining provenance information can become extremely costly, particularly when considering (historic) relationships between multiple objects. Architectural decisions have therefore been taken in each case as to whether the assessment is carried out by the information-centric network or left to the application.

An additional consideration for authentication is whether a name should be irrevocably and immutably tied to a static piece of preexisting content or whether the name can be used to refer to dynamically or subsequently generated content. Schemes that only target immutable content can be less resource hungry as they can use digest functions rather than public key cryptography for generating and checking signatures. However, this can increase the load on applications because they are required to manage many names, rather than using a single name for an item of evolving content that changes over time (e.g., a piece of data containing an age reference).

DONA (Data Oriented Network Architecture) [DONA] and CCN [CCN] [SECCONT] integrate most of the data needed to verify provenance into all content retrievals but need to be able to retrieve additional information (typically a security certificate) in order to complete the provenance authentication. Whether the application has any control of this extra retrieval will depend on the implementation. CCN is explicitly designed to handle dynamic content allowing names to be pre-allocated and attached to subsequently generated content. DONA offers variants for dynamic and immutable content.

PURSUIT [PSTSEC] appears to allow implementers to choose the authentication mechanism so that it can, in theory, emulate the authentication strategy of any of the other architectures. It is not clear whether different choices would lead to lack of interoperability.

NetInf uses the Named Information (ni) URI scheme [RFC6920] to identify content. This allows NetInf to assure validity without any additional information but gives no assurance on provenance or relevance. A "search" request allows an application to identify relevant content and applications may choose to structure content to allow provenance assurance but this will typically require additional network access. NetInf validity authentication is consequently efficient in a network environment with intermittent connectivity as it does not force additional network accesses and allows the

application to decide on provenance validation if required. For dynamic content NetInf can use e.g. signed manifests. For more details on NetInf security see [SCNETINF].

3.2. Authorization, Access Control and Logging

A potentially major concern for all ICN architectures considered here is that they do not provide any inbuilt support for an authorization framework or for logging. Once content has been published and cached in servers, routers or end points not controlled by the publisher, the publisher has no way to enforce access control, determine which users have accessed the content or revoke its publication. In fact, in some cases, it is even difficult for the publishers themselves to perform access control, where requests do not necessarily contain host/user identifier information.

Access could be limited by encrypting the content but the necessity of distributing keys out-of-band appears to negate the advantages of in-network caching. This also creates significant challenges when attempting to manage and restrict key access. An authorization delegation scheme has been proposed [ACDICN]. This scheme allows semi-trusted entities (such as caches, CDN nodes) to delegate access control decisions to 3rd party access control providers that are trusted by the content publisher. The former entities have no access to subscriber related information and should respect the decisions of the access control providers.

A recent proposal for an extra layer in the protocol stack [LIRA] gives control of the name resolution infrastructure to the publisher. This enables access logging as well some degree of active cache management, e.g., purging of stale content.

One possible technique that could allow for providing access control to heterogeneous groups and still allow for a single encrypted object representation that remains cacheable is Attribute Based Encryption (ABE). A first proposal for this is presented in [ABE]. To support heterogeneous group and avoid having a single authority that has a master key multi authority ABE can be used [LEWKO].

Evaluating the impact of the absence of these features will be essential for any scenario where an ICN architecture might be deployed. It may have a seriously negative impact on the applicability of ICN in commercial environments unless a solution can be found.

3.3. Privacy

Another area where the architectures have not been significantly analyzed is privacy. Caching implies a trade-off between network efficiency and privacy. The activity of users is significantly more exposed to the scrutiny of cache owners with whom they may not have any relationship. However it should be noted that it is only the first hop router/cache that can see who request what, this as requests are aggregated and only the previous hop router is visible when a request is forwarded.

Although in many ICN architectures the source of a request is not explicitly identified, an attacker may be able to obtain considerable information if s/he can monitor transactions on the cache and obtain details of the objects accessed, the topological direction of requests and information about the timing of transactions. The persistence of data in the cache can make life easier for an attacker by giving a longer timescale for analysis.

The impact of CCN on privacy has been investigated in [CCNSEC] and the analysis is applicable to all ICN architectures because it is mostly focused on the common caching aspect. The privacy risks of named data networking are also highlighted in [CCNPRIV]. Further work on privacy in ICNs can be found in [CONPRV]. Finally, Fotiou et al. define an ICN privacy evaluation framework in [PRIFRA].

3.4. Changes to the Network Security Threat Model

The architectural differences of the various ICN models as compared to TCP/IP have consequences for network security. There is limited consideration of the threat models and potential mitigation in the various documents describing the architectures. [CCNSEC] and [CONPRV] also consider the changed threat model. Some of the key aspects are:

- Caching implies a tradeoff between network efficiency and user privacy as discussed in Section 4.3.
- More powerful routers upgraded to handle persistent caching increase the network’s attack surface. This is particularly the case in systems that may need to perform cryptographic checks on content that is being cached. For example, not doing this could lead routers to disseminate invalid content.
- ICNs makes it difficult to identify the origin of a request as mentioned in Section 4.3 slowing down the process of blocking requests and requiring alternative mechanisms to differentiate legitimate requests from inappropriate ones as access control lists (ACLs) will probably be of little value for ICN requests.
- Denial-of-service (DoS) attacks may require more effort on ICN

than on TCP/IP but they are still feasible. One reason for this is that it is difficult for the attacker to force repeated requests for the same content onto a single node; ICNs naturally spread content so that after the initial few requests, subsequent requests will generally be satisfied by alternative sources, blunting the impact of a DoS attack. That said, there are many ways around this, e.g., generating random suffix identifiers that always result in cache misses.

- Per-request state in routers can be abused for DoS attacks.
- Caches can be misused in the following ways:
  - Attackers can use caches as storage to make their own content available.
  - The efficiency of caches can be decreased by attackers with the goal of DoS attacks.
  - Content can be extracted by any attacker connected to the cache, putting users' privacy at risk.

Appropriate mitigation of these threats will need to be considered in each scenario.

4. Evaluation Tools

Since ICN is an emerging area, the community is in the process of developing effective evaluation environments, including releasing open-source implementations, simulators, emulators, and testbeds. To date, none of the available evaluation tools can be seen as the one and only community reference evaluation tool. Furthermore, no single environment supports all well-known ICN approaches, as we describe below, hindering the direct comparison of the results obtained for different ICN approaches. The rest of this subsection reviews the publicly available ICN implementations, simulators and experimental facilities currently available to the community.

An updated version of the information in this section will be maintained at the ICNRG Wiki page: https://trac.tools.ietf.org/group/irtf/trac/wiki/IcnEvaluationAndTestbeds

4.1. Open-source Implementations

The Named Data Networking (NDN) project has open-sourced a software reference implementation of the architecture and protocol called NDN (http://http://named-data.net). NDN is available for deployment on

various operating systems and includes C and Java libraries that can be used to build applications.

CCN-lite (http://www.ccn-lite.net) is a lightweight implementation of the CCN protocol that supports most of the key features of CCNx and is interoperable with CCNx. CCN-lite implements the core CCN logic in about 1000 lines of code, so it is ideal for classroom work and course projects as well as for quickly experimenting with CCN extensions. For example, Baccelli et al. use CCN-lite on top of the RIOT operating system to conduct experiments over an IoT (Internet of Things) testbed [NDNIOT].

PARC is offering CCN source code under various licensing schemes, please see http://www.ccnx.org for details.

The PURSUIT project (http://www.fp7-pursuit.eu) has open-sourced its Blackhawk publish-subscribe (Pub/Sub) implementation for Linux and Android; more details are available at https://github.com/fp7-pursuit/blackadder. Blackadder uses the Click modular router for ease of development. The code distribution features a set of tools, test applications and scripts. The POINT project (http://www.point-h2020.eu/) is currently maintaining Blackadder.

The 4WARD and SAIL projects have open-sourced software that implements different aspects of NetInf, e.g., NetInf URI format, HTTP and UDP convergence layer, using different programming languages. The Java implementation provides a local caching proxy and client. Further, an OpenNetInf prototype is available as well as a hybrid host-centric and information-centric network architecture called the Global Information Network (GIN), a browser plug-in and video streaming software. See http://www.netinf.org/open-source for more details.

4.2. Simulators and Emulators

Simulators and emulators should be able to capture faithfully all features and operations of the respective ICN architecture(s) and any limitations should be openly documented. It is essential that these tools and environments come with adequate logging facilities so that one can use them for in-depth analysis as well as debugging. Additional requirements include the ability to support medium- to large-scale experiments, the ability to quickly and correctly set various configurations and parameters, as well as to support the playback of traffic traces captured on a real testbed or network. Obviously, this does not even begin to touch upon the need for strong validation of any evaluated implementations.

4.2.1. ndnSIM

The Named Data Networking (NDN) project (http://named-data.net/) has developed ndnSIM [ndnSIM][ndnSIM2]; this is a module that can be plugged into the ns-3 simulator (https://www.nsnam.org/) and supports the core features of NDN. One can use ndnSIM to experiment with various NDN applications and services as well as components developed for NDN such as routing protocols, caching and forwarding strategies among others. The code for ns-3 and ndnSIM is openly available to the community and can be used as the basis for implementing ICN protocols or applications. For more details see http://ndnsim.net/2.0/.

4.2.2. ccnSIM

ccnSim [ccnSim] is CCN-specific simulator that was specially designed to handle forwarding of a large number of CCN-chunks (http://www.infres.enst.fr/~drossi/index.php?n=Software.ccnSim). ccnSim is written in C++ for the OMNeT++ simulation framework (https://omnetpp.org/). Other CCN-specific simulators include CCN Packet Level Simulator [CCNPL] and CCN-Joker [CCNj]. CCN-Joker emulates in user-space all basic aspects of a CCN node (e.g., handling of Interest and Data packets, cache sizing, replacement policies), including both flow and congestion control. The code is open source and is suitable for both emulation-based analyses and real experiments. Finally, Cabral et al. [MiniCCNx] use container-based emulation and resource isolation techniques to develop a prototyping and emulation tool.

4.2.3. Icarus simulator

The Icarus simulator focuses on caching in ICN and is agnostic with respect to any particular ICN implementation. The simulator is implemented in Python, uses the Fast Network Simulator Setup tool [FNSS], and is available at http://icarus-sim.github.io/. Icarus has several caching strategies implemented, including among others ProbCache [PROBCACH], node-centrality-based caching [CL4M] and hash-route-based caching [HASHROUT].

ProbCache [PROBCACH] is taking a resource management view on caching decisions and approximates the available cache capacity along the path from source to destination. Based on this approximation and in order to reduce caching redundancy across the path, it caches content probabilistically. According to [CL4M] the node with the highest betweenness centrality along the path from source to destination is responsible for caching incoming content. Finally, [HASHROUT] calculates the hash-function of a content’s name and assigns contents

to caches of a domain according to that. The hash-space is split according to the number of caches of the network. Then upon subsequent requests, and based again on the hash of the name included in the request, edge routers redirect requests to the cache assigned with the corresponding hash-space. [HASHROUT] is an off-path caching strategy, in contrast to [PROBCACH] and [CL4M], which however, requires minimum co-ordination and redirection overhead. In its latest update, Icarus also includes implementation of the "Satisfied Interest Table" (SIT) [SIT]. The SIT table is pointing towards the direction where content has been sent recently. Among other benefits, this enables information resilience in case of network fragmentation (i.e., content can still be found in neighbour caches or in users' devices) and inherently supports user-assisted caching (i.e., P2P-like content distribution).

The simulator is constantly being updated and more strategies are added in each update. Tortelli et al. [ICNSIMS] provide a comparison of ndnSIM, ccnSim, and Icarus.

4.3. Experimental Facilities

An important consideration in the evaluation of any kind of future Internet mechanism lies in the characteristics of that evaluation itself. Central to the assessment of the features provided by a novel mechanism, lies the consideration of how it improves over already existing technologies, and by "how much." With the disruptive nature of clean-slate approaches generating new and different technological requirements, it is complex to provide meaningful results for a network layer framework, in comparison with what is deployed in the current Internet. Thus, despite the availability of ICN implementations and simulators, the need for large-scale environments supporting experimental evaluation of novel research is of prime importance to the advancement of ICN deployment.

Different experimental facilities have different characteristics and capabilities, e.g. low cost of use, reproducible configuration, easy-to-use tools, sharable, available background traffic.

4.3.1. Open Network Lab [ONL]

An example of an experimental facility that supports CCN is the Open Network Lab [ONL] that currently comprises 18 extensible gigabit routers and over a 100 computers representing clients and is freely available to the public for running CCN experiments. Nodes in ONL are preloaded with CCNx software. ONL provides a graphical user interface for easy configuration and testbed setup as per the experiment requirements, and also serves as a control mechanism, allowing access to various control variables and traffic counters.

Further, it is also possible to run and evaluate CCN over popular testbeds [PLANET] [EMULAB] [DETERLAB] [OFELIA] by directly running, for example, the CCNx open-source code [CCNOFELI] [ICNGRID] [CCNPL] [CCNOSN]. Also, the Network Experimentation Programming Interface [NEPI] is a tool developed for controlling and managing large-scale network experiments. NEPI can be used to control and manage large-scale CCNx experiments e.g., on PlanetLab [NEPIICN].

4.3.2. POINT testbed

The POINT project is maintaining a testbed with 40 machines across Europe, North America (MIT) and Japan (NICT) interconnected in a topology containing one Topology Manager and one Rendezvous node that handle all publish/subscribe and topology formation requests [IEICE]. All machines run Blackadder. New nodes can join and experiments can be run on request.

4.3.3. CUTEi: Container-based ICN testbed

NICT has also developed a testbed used for ICN experiments [AFI] comprising multiple servers located in Asia and other locations. Each testbed server (or VM) utilizes a Linux kernel-based container (LXC) for node virtualization. This testbed enables users to run applications and protocols for ICN in two experimentation modes using two different container designs:

1. application-level experimentation using a "common container" and

2. network-level experimentation using a "user container."

A common container is shared by all testbed users, and a user container is assigned to one testbed user. A common container has a global IP address to connect with other containers or external networks, whereas each user container uses a private IP address and a user space providing a closed networking environment. A user can login to his/her user containers using SSH with his/her certificate, or access them from PCs connected to the Internet using SSH tunnelling.

This testbed also implements an "on-filesystem cache" to allocate caching data on a UNIX filesystem. The on-filesystem cache system accommodates two kinds of caches: "individual cache" and "shared cache." Individual cache is accessible for one dedicated router for the individual user, while shared cache is accessible for a set of routers in the same group to avoid duplicated caching in the neighborhood for cooperative caching.

5. Security Considerations

This document does not impact the security of the Internet.

6. IANA Considerations

This document presents no IANA considerations.

7. Acknowledgments

Konstantinos Katsaros contributed the updated text of Section 2.2 along with an extensive set of references.

Priya Mahadevan, Daniel Corujo and Gareth Tyson contributed to an earlier version of this document.

This document has benefited from reviews, pointers to the growing ICN literature, suggestions, comments and proposed text provided by the following members of the IRTF Information-Centric Networking Research Group (ICNRG), listed in alphabetical order: Marica Amadeo, Hitoshi Asaeda, E. Baccelli, Claudia Campolo, Christian Esteve Rothenberg, Suyong Eum, Nikos Fotiou, Dorothy Gellert, Luigi Alfredo Grieco, Myeong-Wuk Jang, Ren Jing, Will Liu, Antonella Molinaro, Luca Muscariello, Ioannis Psaras, Dario Rossi, Stefano Salsano, Damien Saucez, Dirk Trossen, Jianping Wang, Yuanzhe Xuan, and Xinwen Zhang.

The IRSG review was provided by Aaron Falk.

8. Informative References

[RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, May 1998.

[RFC5743] Falk, A., "Definition of an Internet Research Task Force (IRTF) Document Stream", RFC 5743, December 2009.

[RFC6920] Farrell, S., Kutscher, D., Dannewitz, C., Ohlman, B., Keranen, A., and P. Hallam-Baker, "Naming Things with Hashes", RFC 6920, April 2013.

[RFC7476] Pentikousis, K., Ohlman, B., Corujo, D., Boggia, G., Tyson, G., Davies, E., Molinaro, A., and S. Eum, "Information-Centric Networking: Baseline Scenarios ", RFC

7476, March 2015.

[ICNCHA] Internet draft, "ICN Research Challenges" draft-irtf-icnrg-challenges-04, [***EDITORS NOTE: This reference is expected to have become an RFC by the time this document is published, this reference should thus be updated before publication of this document.***]

[ndnSIM] Afanasyev, A. et al., "ndnSIM: NDN simulator for NS-3", NDN Technical Report NDN-0005, Revision 2, October 2012.

[ndnSIM2] Mastorakis, S. et al., "ndnSIM 2.0: A new version of the NDN simulator for NS-3", NDN Technical Report NDN-0028, Revision 1, January 2015.

[ccnSim] Rossini, G. and D. Rossi, "Large scale simulation of CCN networks", Proc. Algotel 2012 , La Grande Motte, France, May 2012.

[CCNPL] Muscariello, L., "Content centric networking packet level simulator", available online at http://perso.rd.francetelecom.fr/muscariello/sim.html

[CCNj] Cianci, I. et al. "CCN - Java Opensource Kit EmulatoR for Wireless Ad Hoc Networks", Proc. 7th ACM Int. Conf. on Future Internet Technologies, Seoul, Korea, Sept., 2012.

[IEICE] G. Parisis, D. Trossen, and H. Asaeda, "A Node Design and a Framework for Development and Experimentation for an Information-Centric Network", IEICE Trans. Commun., vol. E96-B, no. 7, pp.1650-1660, July 2013.

[PROBCACH] I. Psaras, W. Chai, G. Pavlou, "Probabilistic In-Network Caching for Information-Centric Networks", Proc. SIGCOMM ICN Workshop. ACM, 2012.

[CL4M] Chai, W. K. et al., "Cache ’Less for More’ in Information-centric Networks", Proc. Networking. IFIP, 2012.

[SIT] Vasilis Sourlas, Leandros Tassiulas, Ioannis Psaras and George Pavlou, "Information Resilience through User-Assisted Caching in Disruptive Content-Centric Networks" 14th IFIP NETWORKING, Toulouse, France, May 2015.

[HASHROUT] L. Saino, I. Psaras, G. Pavlou, "Hash-routing Schemes for Information-Centric Networking", Proc. SIGCOMM ICN Workshop. ACM, 2013.

[ICARUS] L. Saino, I. Psaras, G. Pavlou, "Icarus: a Caching Simulator for Information Centric Networking (ICN)", Proc. SIMUTOOLS. ICST, 2014.

[FNSS] L. Saino, C. Cocora and G. Pavlou, "A Toolchain for Simplifying Network Simulation Setup", Proc. SIMUTOOLS. ACM, 2013.

[BA] Barabasi, A. and R. Albert, "Emergence of scaling in random networks", Science, vol. 286, no. 5439, pp. 509-512, 1999.

[WATTS] Watts, D. J. and S. H. Strogatz, "Collective dynamics of small-world networks", Nature, vol. 393, no. 6684, pp. 40-"10, 1998.

[Montage] Hussain, A. and J. Chen, "Montage Topology Manager: Tools for Constructing and Sharing Representative Internet Topologies", DETER Technical Report, ISI-TR-684, Aug. 2012.

[DELAY] Kaune, S. et al., "Modelling the Internet Delay Space Based on Geographical Locations", Proc. Euromicro, Weimar, Germany, 2009.

[IMB2014] C. Imbrenda, L. Muscariello, and D. Rossi,"Analyzing Cacheable Traffic in ISP Access Networks for Micro CDN Applications via Content-Centric Networking," In Proc. of ACM ICN, 2014.

[L1] http://googleblog.blogspot.it/2008/07/we-knew-web-was-big.html

[L2] Zhang, C., Dhungel, P., and K. Di Wu., "Unraveling the BitTorrent ecosystem", IEEE Transactions on Parallel and Distributed Systems, pp. 1164-1177, 2010.

[L3] Cha, M., Kwak, H., Rodriguez, P., Ahn, Y.-Y., and S. Moon, "I tube, you tube, everybody tubes: analyzing the world’s largest user generated content video system", Proc. ACM SIGCOMM conference on Internet measurement (IMC), San Diego (CA), USA, Oct. 2007.

[L4] Zhou, J., Li, Y., Adhikari, K., and Z.-L. Zhang, "Counting YouTube videos via random prefix sampling", In Proc. of IMC’11, Berlin, Germany, Nov. 2011.

[L5] Fricker, C., Robert, P., Roberts, J. and N. Sbihim,

"Impact of traffic mix on caching performance in a content-centric network", In Proc. of IEEE NOMEN 2012, Workshop on Emerging Design Choices in Name-Oriented Networking, Orlando, USA, Mar. 2012.

[L6] Yu, H., Zheng, D., Zhao, B. Y. and W. Zheng, "Understanding user behavior in large-scale video-on-demand systems", In SIGOPS Oper. Syst. Rev., Vol. 40, pp. 333-344, April 2006.

[L7] Marciniak, P., Liogkas, N., Legout, A. and E. Kohler, "Small is not always beautiful", In Proc. of IPTPS, International Workshop of Peer-to-Peer Systems, Tampa Bay, Florida (FL), USA, Feb. 2008.

[L8] Bellissimo, A., Levine, B. and P. Shenoy, "Exploring the use of BitTorrent as the basis for a large trace repository", University of Massachusetts, Tech. Rep., 2004.

[L9] Psaras, I. et al., "Modelling and Evaluation of CCN-Caching Trees", In Proc. of the 10th international IFIP conference on Networking, Valencia, Spain, May 2011.

[L10] Carofiglio, G., Gallo, M., Muscariello, L., and D. Perino, "Modeling Data Transfer in Content-Centric Networking", In Proc. of ITC, San Francisco, USA, Sep. 2011.

[L11] Breslau, L., Cao, P., Fan, L., Phillips, G. and S. Shenker, "Web caching and zipf-like distributions: evidence and implications", In Proc. of INFOCOM ’99, New York (NY), USA, Mar. 1999.

[L12] Mahanti, A., Williamson, C., and D. Eager., "Traffic analysis of a web proxy caching hierarchy", IEEE Network, Vol.14, No.3, pp.16-23, May/June 2000.

[P2PMod] Saleh, O., and M. Hefeeda, "Modeling and caching of peer-to-peer traffic", Proc. ICNP. IEEE, 2006.

[W16] C. Westphal, Ed., "Adaptive Video Streaming over ICN", ICNRG Internet Draft.

[Led12] S. Lederer, C. Muller, and C. Timmerer, "Dynamic adaptive streaming over HTTP dataset", in Proceedings of the ACM Multimedia Systems Conference 2012, 2012, pp. 89-94.

[CCN] Jacobson, V. et al., "Networking Named Content", Proc.

CoNEXT. ACM, 2009.

[VoCCN] Jacobson, V. et al., "VoCCN: Voice-over Content-Centric Networks", Proc. CoNEXT Re-Arch Workshop. ACM, 2009.

[NDNP] Zhang, L. et al., "Named Data Networking (NDN) Project", NDN Technical Report NDN-0001, Oct. 2010. Available: http://named-data.net/publications/techreports/

[4WARD6.1] Ohlman, B. et al., "First NetInf Architecture Description", 4WARD Project Deliverable D-6.1, Apr. 2009.

[4WARD6.3] Ahlgren, B. et al., "NetInf Evaluation", 4WARD Project Deliverable D-6.3, June 2010.

[SAIL-B2] SAIL, "NetInf Content Delivery and Operations", SAIL Project Deliverable D-B.2, May. 2012.

[SAIL-B3] Kutscher, D. (ed.) et al., "Final NetInf Architecture", SAIL Project Deliverable D-B.3 , Jan. 2013. Available: http://www.sail-project.eu/deliverables/

[PRST4.5] Riihijarvi, J. et al., "Final Architecture Validation and Performance Evaluation Report", PURSUIT Project Deliverable D4.5, Jan. 2013.

[CMT-D5.2] B ben, A. et al, "Scalability of COMET System", COMET Project Deliverable D5.2, Feb. 2013.

[CMT-D6.2] Georgiades, M. et al., "Prototype Experimentation and Demonstration", COMET Project Deliverable D6.2, Feb. 2013.

[SHARE] Muscariello, L. et al., "Bandwidth and storage sharing performance in information centric networking", Proc. SIGCOMM ICN Workshop. ACM, 2011.

[RealCCN] Perino, D. et al., "A Reality Check for Content Centric Networking", Proc. SIGCOMM ICN Workshop. ACM, 2011.

[ICN-Web] Detti, A. et al., "Supporting the Web with an Information Centric Network that Routes by Name", Elsevier Computer Networks, vol. 56, no. 17, Nov. 2012.

[ICN-Scal] Blefari Melazzi, N. et al., "Scalability Measurements in an Information-Centric Network", Springer Lecture Notes in Computer Science (LNCS), vol. 7586, 2012.

[ICN-Tran] Salsano, S. et al., "Transport-Layer Issues in Information

Centric Networks ", Proc. SIGCOMM ICN Workshop. ACM, 2012.

[SensReqs] Karnouskos, S. et al., "Requirement considerations for ubiquitous integration of cooperating objects", Proc. NTMS. IFIP, 2011.

[NETINFSC] Renault, E, Ahmad, A., and M. Abid, "Towards a Security Model for the Future Network of Information", Proc. Conf. Ubiquitous Information Technologies and Applications, IEEE, 2009.

[DONA] Koponen, T. et al., "A Data-Oriented (and Beyond) Network Architecture", Proc. SIGCOMM. ACM, 2007.

[SECCONT] Smetters, D., and V. Jacobson, "Securing network content", Technical Report TR-2009-01, PARC, 2009.

[PSTSEC] Tagger, B., et al, "Update on the Architecture and Report on Security Analysis", Deliverable 2.4, PURSUIT EU FP7 project, April 2012.

[ABE] Mihaela Ion, Jianqing Zhang, and Eve M. Schooler. 2013. Toward content-centric privacy in ICN: attribute-based encryption and routing. In Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM (SIGCOMM '13). ACM, New York, NY, USA, 513-514. DOI=10.1145/2486001.2491717 http://doi.acm.org/10.1145/2486001.2491717

[LEWKO] Lewko, A., Waters, B.: Decentralizing attribute-based encryption. In: Proc. of EUROCRYPT 11, LNCS, vol. 6632, pp. 568-588 (2011).

[SCNETINF] C. Dannewitz, J. Golic, B. Ohlman, B. Ahlgren, "Secure Naming for A Network of Information," Proc. 13th IEEE Global Internet Symp. 2010, San Diego, CA, Mar. 2010.

[ACDICN] Fotiou, N. et al., "Access control enforcement delegation for information-centric networking architectures", Proc. SIGCOMM ICN Workshop. ACM, 2012.

[CCNSEC] Lauinger, T., "Security and Scalability of Content-Centric Networking", Masters Thesis, Technische Universitaet Darmstadt and Eurecom, Sep. 2010.

[CCNPRIV] Lauinger, Y., et al, "Privacy Risks in Named Data Networking: What is the Cost of Performance?", ACM SIGCOMM Computer Communication Review Editorial Note, vol. 42, iss. 5, 2012

[CONPRV] Chaabane, A et al, "Privacy in Content-Oriented Networking: Threats and Countermeasures", arXiv:1211.5183, 2012.

[1] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian. 2010. Internet inter-domain traffic. In Proceedings of the ACM SIGCOMM 2010 conference (SIGCOMM '10). ACM, New York, NY, USA, 75-86. DOI=10.1145/1851182.1851194, http://doi.acm.org/10.1145/1851182.1851194

[2] G. Maier, A. Feldmann, V. Paxson, and M. Allman. 2009. On dominant characteristics of residential broadband internet traffic. In Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference (IMC '09). ACM, New York, NY, USA, 90-102. DOI=10.1145/1644893.1644904 http://doi.acm.org/10.1145/1644893.1644904

[3] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web Caching and Zipf-like Distributions: Evidence and Implications," in IEEE INFOCOM, 1999, pp. 126-134.

[4] A. Mahanti, C. Williamson, and D. Eager, "Traffic analysis of a web proxy caching hierarchy," IEEE Network, vol. 14, no. 3, pp. 16 -23, 2000.

[5] M. Busari and C. Williamson, "ProWGen: a synthetic workload generation tool for simulation evaluation of web proxy caches," Computer Networks, vol. 38, no. 6, pp. 779-794, 2002.

[6] M. F. Arlitt and C. L. Williamson, "Internet web servers: workload characterization and performance implications," IEEE/ACM Transactions on Networking, vol. 5, pp. 631-645, 1997.

[7] P. Barford and M. Crovella, "Generating representative web workloads for network and server performance evaluation," in ACM SIGMETRICS/PERFORMANCE, 1998, pp. 151-160.

[8] P. Barford, A. Bestavros, A. Bradley, and M. Crovella, "Changes in web client access patterns: Characteristics and caching implications," World Wide Web, vol. 2, pp. 15-28, 1999.

[9] M. Hefeeda and O. Saleh, "Traffic Modeling and Proportional Partial Caching for Peer-to-Peer Systems," IEEE/ACM Transactions on Net- working, vol. 16, no. 6, pp.

1447-1460, 2008.

[10] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang, "A performance study of BitTorrent-like peer-to-peer systems," IEEE Journal on Selected Areas in Communication, vol. 25, no. 1, pp. 155-169, 2007.

[11] A. Bellissimo, B. N. Levine, and P. Shenoy, "Exploring the use of BitTorrent as the basis for a large trace repository," University of Massachusetts Amherst, Tech. Rep., 2004.

[12] X. Cheng, C. Dale, and J. Liu, "Statistics and social network of YouTube videos," in IEEE IWQoS. IEEE, 2008, pp. 229-238.

[13] X. Cheng, C. Dale, and J. Liu, "Understanding the Characteristics of Internet Short Video Sharing: YouTube as a Case Study," CoRR, vol. abs/0707.3670, 2007.

[14] K. V. Katsaros, G. Xylomenos, and G. C. Polyzos, "GlobeTraff: a traffic workload generator for the performance evaluation of future Internet architectures," Proc. IEEE/IFIP NTMS, 2012.

[MiniCCNx] C. Cabral, et al., "High fidelity content-centric experiments with Mini-CCNx", Proc. ISCC. IEEE, 2014.

[ICNSIMS] M. Tortelli, et al., "CCN Simulators: Analysis and Cross-Comparison", Proc. SIGCOMM ICN. ACM, 2014.

[AFI] H. Asaeda, R. Li, and N. Choi, "Container-Based Unified Testbed for Information-Centric Networking," IEEE Network, vol. 28, no. 6, pp. 60-66, 2014.

[NDNIOT] E. Baccelli, et al., "Information Centric Networking in the IoT: Experiments with NDN in the Wild," Proc. SIGCOMM ICN. ACM, 2014.

[ONL] J. DeHart, et al., "The open network laboratory: a resource for networking research and education", ACM SIGCOMM CCR, vol. 35, no. 5, pp. 75-78, 2005.

[PLANET] B. Chun, et al., "Planetlab: an overlay testbed for broad-coverage services", ACM SIGCOMM CCR, vol. 33, no. 3, pp. 3-12, 2003.

[EMULAB] E. Eide, et. al., "An Experimentation Workbench for Replayable Networking Research", Proc. NSDI. USENIX, 2007.

[DETERLAB] T. Benzel, "The Science of Cyber-Security Experimentation: The DETER Project", Proc. Annual Computer Security Applications Conference (ACSAC), Dec. 2011.

[OFELIA] M. Sune, et al., "Design and implementation of the OFELIA FP7 facility: The European OpenFlow testbedDesign and implementation of the OFELIA FP7 facility: The European OpenFlow testbed", Computer Networks, vol. 61, pp. 132-150, March 2014.

[NEPI] A. Quereilhac, et al., "NEPI: An integration framework for Network Experimentation", Proc. SoftCOM. IEEE, 2011.

[NEPIICN] A. Quereilhac, et al., "Demonstrating a unified ICN development and evaluation framework", Proc. SIGCOMM ICN. ACM, 2014.

[CCNOFELI] S. Salsano, et al., "Information Centric Networking over SDN and OpenFlow: Architectural Aspects and Experiments on the OFELIA Testbed", Computer Networks, vol. 57, no. 16, pp. 3207-3221, November 2013.

[ICNGRID] G. Carofiglio, et al., "Optimal multipath congestion control and request forwarding in Information-Centric Networks", Proc. ICNP. IEEE, 2013.

[CCNPL] S. Awiphan, et al., "Video streaming over content centric networking: Experimental studies on PlanetLab", Proc. ComComAp. IEEE, 2013

[CCNOSN] C. Bernardini, et al., "Socially-aware caching strategy for content centric networking", Proc. Networking. IFIP, 2014.

[PRIFRA] N. Fotiou, et al. "A framework for privacy analysis of ICN architectures", Proc. APF. Springer, 2014.

[ICNScale] K. V. Katsaros, et al., "On the Inter-domain Scalability of Route-by-Name Information-Centric Network Architectures", Proc. Networking. IFIP, 2015.

[COMPLEX] X. Dimitropoulos, et al., "Graph annotations in modeling complex network topologies", ACM Trans. Model. Comput. Simul., vol. 19, no. 4, Nov. 2009.

[LIRA] I. Psaras, K. Katsaros, L. Saino, and G. Pavlou, "Lira: A location independent routing layer based on source-provided ephemeral names," Electronic and Electrical Eng. Dept., UCL, London, UK, Tech. Rep. 2014. [Online]. Available: http://www.ee.ucl.ac.uk/comit-project/publications.html

Authors' Addresses

Kostas Pentikousis (editor)
EICT GmbH
Torgauer Strasse 12-15
10829 Berlin
Germany

Email: k.pentikousis@eict.de

Borje Ohlman
Ericsson Research
S-16480 Stockholm
Sweden

Email: Borje.Ohlman@ericsson.com

Elwyn Davies
Trinity College Dublin/Folly Consulting Ltd
Dublin, 2
Ireland

Email: davieseb@scss.tcd.ie

Spiros Spirou
Intracom Telecom
19.7 km Markopoulou Avenue
19002 Peania, Athens
Greece

Email: spis@intracom-telecom.com

Gennaro Boggia
Dep. of Electrical and Information Engineering
Politecnico di Bari
Via Orabona 4

70125 Bari
Italy

Email: g.boggia@poliba.it

ICNRG
Internet Draft
Intended status: Informational
Expires: October 26, 2016

C. Westphal, Ed.
Huawei
S. Lederer
D. Posh
C. Timmerer
Alpen-Adria University Klagenfurt
Aytac Azgin
S. Liu
Huawei
C. Mueller
Bitmovin
A.Detti
University of Rome Tor Vergata
D. Corujo
University of Aveiro
J. Wang
City University of Hong-Kong
Marie-Jose Montpetit
Niall Murray
Athlone Institute of Technology

April 27, 2016

Adaptive Video Streaming over ICN
draft-irtf-icnrg-videostreaming-08.txt

Status of this Memo

Copyright Notice

Abstract

This document considers the consequences of moving the underlying network architecture from the current Internet to an Information-Centric Network (ICN) architecture on video distribution. As most of the traffic in future networks is expected to be video, we consider how to modify the existing video streaming mechanisms. Several important topics related to video distribution over ICN are presented, covering a wide range of scenarios: we look at how to evolve DASH to work over ICN, and leverage the recent ISO/IEC Moving Picture Experts Group (MPEG) Dynamic Adaptive Streaming over HTTP (DASH) standard; we consider layered encoding over ICN; Peer-to-Peer

(P2P) mechanisms introduce distinct requirements for video and we look at how to adapt PPSP for ICN; Internet Protocol Television (IPTV) adds delay constraints, and this will create more stringent requirements over ICN as well. As part of the discussion on video, we discuss Digital Rights Management (DRM) in ICN. Finally, in addition to considering how existing mechanisms would be impacted by ICN, this document lists some research issues to design ICN specific video streaming mechanisms.

Table of Contents

1. Introduction

The unprecedented growth of video traffic has triggered a rethinking of how content is distributed, both in terms of the underlying Internet architecture and in terms of the streaming mechanisms to deliver video objects.

In particular, the IRTF ICNRG research group has been chartered to study new architectures centered upon information; the main contributor to Internet traffic (and information dissemination) is video, and this is expected to stay the same in the near future. If ICN is expected to become prominent, it will have to support video streaming efficiently.

As such, it is necessary to discuss along two directions:

- Can the current video streaming mechanisms be leveraged and adapted to an ICN architecture?

- Can (and should) new, ICN-specific video streaming mechanisms be designed to fully take advantage of the new abstractions exposed by the ICN architecture?

This document intends to focus on the first question, in an attempt to define the use cases for video streaming and some requirements.

This document focuses on a few scenarios, namely Netflix-like video streaming, peer-to-peer video sharing and IPTV, and identifies how the existing protocols can be adapted to an ICN architecture. In doing so, it also identifies the main issues with these protocols in this ICN context.

Some documents have started to consider the ICN-specific requirements of dynamic adaptive streaming [2][3][4][6].

In this document, we give a brief overview of the existing solutions for the selected scenarios. We then consider the interactions of such existing mechanisms with the ICN architecture and list some of the interactions any video streaming mechanism will have to consider. We then identify some areas for future research.

2. Conventions used in this document

In examples, "C:" and "S:" indicate lines sent by the client and server respectively.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [RFC2119].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying RFC-2119 significance.

3. Use case scenarios for ICN and Video Streaming

For ICN specific descriptions, we refer to the other research group documents. For our purpose, we assume here that ICN means an architecture where content is retrieved by name and with no binding of content to a specific network location.

The consumption of multimedia content comes along with timing requirements for the delivery of the content, for both, live and on-demand consumption. Additionally, real-time use cases such as audio-/video conferencing [7], game streaming, etc., come along with more strict timing requirements. Long startup delays, buffering periods or poor quality, etc., should be avoided to achieve a good Quality of Experience (QoE) to the consumer of the content. (For a definition of QoE in the context of video distribution, please refer to [25]. The working definition is: "Quality of Experience (QoE) is the degree of delight or annoyance of the user of an application or service. It results from the fulfillment of his or her expectations

with respect to the utility and / or enjoyment of the application or service in the light of the user's personality and current state.")

Of course, these requirements are heavily influenced by routing decisions and caching, which are central parts of ICN and which have to be considered when streaming video in such infrastructures.

Due to this range of requirements, we find it useful to narrow the focus on four scenarios (more can be included later):

- a video download architecture similar to that of Apple iTtunes(R), where the whole file is being downloaded to the client and can be replayed there multiple times;
- a video streaming architecture for playing back movies; this is relevant for the naming and caching aspects of ICN, as well as the interaction with the rate adaptation mechanism necessary to deliver the best QoE to the end-user;
- a peer-to-peer architecture for sharing videos; this introduces more stringent routing requirements in terms of locating copies of the content, as the location of the peers evolves and peers join and leave the swarm they use to exchange video chunks (for Peer-to-Peer definitions and taxonomy, please refer to RFC5694);
- IPTV; this introduces requirements for multicasting and adds stronger delay constraints.

Other scenarios, such as video-conferencing and real-time video communications are not explicitly discussed in this document, while they are in scope. Also, events of mass-media distribution, such as a large crowd in a live event, are also adding new requirements to be included in later version.

We discuss how the current state-of-the-art protocols in an IP context can be modified for the ICN architecture. The remainder of this document is organized as follows. In the next section, we consider video download. Then in Section 5, we briefly describe DASH [1], and Layered Encoding (MDC, SVC). P2P is the focus of Section 6, where we describe PPSP. Section 7 highlights the requirements of IPTV, while Section 8 describes the issues of DRM. Section 9 lists some research issues to be solved for ICN-specific video delivery mechanisms.

Videoconferencing and real-time video communications will be detailed more in future versions of this document; as well as the mass distribution of content at live large-scale events (stadium, concert hall, etc) for which there is no clearly adopted existing protocol.

4. Video download

Video download, namely the fetching of a video file from a server or a cache down to the user's local storage, is a natural application of ICN. It should be supported natively without requiring any specific considerations.

This is supported now by a host of protocols (say, SCP, FTP, or over HTTP), which would need to be replaced by the protocols to retrieve content in ICNs.

However, current mechanisms are built atop existing transport protocols. Some ICN proposals (say, CCN or NDN for instance) attempt to leverage the work done upon these transport protocol and it has been proposed to use mechanisms such as the TCP congestion window (and the associated Adaptive Increase, Multiplicative Decrease - AIMD) to decide how many object requests ("interests" in CCN/NDN terminology) should be in flight at any point in time.

It should be noted that ICN intrinsically supports different transport mechanisms, which could achieve better performance than TCP, as they subsume TCP into a special case. For instance, one could imagine a link-by-link transport coupled with caching. This is enabled by the ICN architecture, and would facilitate the point-to-point download of video files.

5. Video streaming and ICN

5.1. Introduction to client-driven streaming and DASH

Media streaming over the hypertext transfer protocol (HTTP) and in a further consequence streaming over the transmission control protocol (TCP) has become omnipresent in today's Internet. Content providers such as Netflix, Hulu, and Vudu do not deploy their own streaming equipment but use the existing Internet infrastructure as it is and they simply deploy their own services over the top (OTT). This streaming approach works surprisingly well without any particular support from the underlying network due to the use of efficient video compression, content delivery networks (CDNs), and adaptive video players. Earlier video streaming research mostly recommended to use the user datagram protocol (UDP) combined with the real time transport protocol (RTP). It assumed it would not be possible to transfer multimedia data smoothly with TCP, because of its throughput variations and large retransmission delays. This point of view has significantly evolved today. HTTP streaming, and especially its most simple form known as progressive download, has become very popular over the past few years because it has some major benefits

compared to RTP streaming. As a consequence of the consistent use of HTTP for this streaming method, the existing Internet infrastructure, consisting of proxies, caches and CDNs, could be used. Originally, this architecture was designed to support best effort delivery of files and not real time transport of multimedia data. Nevertheless, real time streaming based on HTTP could also take advantage of this architecture, in comparison to RTP, which could not leverage any of the aforementioned components. Another benefit that results from the use of HTTP is that the media stream could easily pass firewalls or network address translation (NAT) gateways, which was definitely a key for the success of HTTP streaming. However, HTTP streaming is not the holy grail of streaming as it also introduces some drawbacks compared to RTP. Nevertheless, in an ICN-based video streaming architecture these aspects also have to be considered.

The basic concept of DASH [1] is to use segments of media content, which can be encoded at different resolutions, bit rates, etc., as so-called representations. These segments are served by conventional HTTP Web servers and can be addressed via HTTP GET requests from the client. As a consequence, the streaming system is pull-based and the entire streaming logic is located on the client, which makes it scalable, and allows to adapt the media stream to the client’s capabilities.

In addition to this, the content can be distributed using conventional CDNs and their HTTP infrastructure, which also scales very well. In order to specify the relationship between the contents’ media segments and the associated bit rate, resolution, and timeline, the Media Presentation Description (MPD) is used, which is a XML document. The MPD refers to the available media segments using HTTP URLs, which can be used by the client for retrieving them.

5.2. Layered Encoding

Another approach for video streaming consist in using layered encoding. Namely, scalable video coding formats the video stream into different layers: a base layer which can be decoded to provide the lowest bit rate for the specific stream, and enhancement layers which can be transmitted separately if network conditions allow. The higher layers offer higher resolutions and enhancement of the video quality, while the layered approach allows to adapt to the network conditions. This is used in MPEG-4 scalable profile or H.263+. H264SVC is available, but not much deployed. JPEG2000 has a wavelet transform approach for layered encoding, but has not been deployed much either.

It is not clear if the layered approach is fine-grained enough for rate control.

5.3. Interactions of Video Streaming with ICN

5.3.1. Interaction of DASH and ICN

Video streaming, and DASH in particular, have been designed with goals that are aligned with that of most ICN proposals. Namely, it is a client-based mechanism, which requests items (in this case, chunks of a video stream) by name.

ICN and MPEG-DASH [1] have several elements in common:

- the client-initiated pull approach;
- the content being dealt with in pieces (or chunks);
- the support of efficient replication and distribution of content pieces within the network;
- the scalable, session-free nature of the exchange between the client and the server at the streaming layer: the client is free to request any chunk from any location;
- the support for potentially multiple source locations.

For the last point, DASH may list multiple source URLs in a manifest, and ICN is agnostic to the location of a copy it is receiving. We do not imply that current video streaming mechanisms attempt to draw the content from multiple sources concurrently. This is a potential benefit of ICN, but is not considered in the current approaches mentioned in this document.

As ICN is a promising candidate for the Future Internet (FI) architecture, it is useful to investigate its suitability in combination with multimedia streaming standards like MPEG-DASH. In this context, the purpose of this section is to present the usage of ICN instead of HTTP in MPEG-DASH

However, there are some issues that arise from using a dynamic rate adaptation mechanism in an ICN architecture (note that some of the issues are related to caching, and not necessarily unique to ICN):

- Naming of the data in DASH does not necessarily follow the ICN convention of any of the ICN proposals. Several chunks of the same video stream might currently go by different names that for instance do not share a common prefix. There is a need to harmonize the naming of the chunks in DASH with the naming conventions of the ICN. The naming convention of using a filename/time/encoding format could for instance be made compatible with the convention of CCN.

- While chunks can be retrieved from any server, the rate adaptation mechanism attempts to estimate the available network bandwidth so as to select the proper playback rate and keep its playback buffer at the proper level. Therefore, there is a need to either include some location semantics in the data chunks so as to properly assess the throughput to a specific location; or to design a different mechanism to evaluate the available network bandwidth.

- The typical issue of access control and accounting happens in this context, where chunks can be cached in the network outside of the administrative control of the content publisher. It might be a requirement from the owner of the video stream that access to these data chunks needs to be accounted/billed/monitored.

- Dynamic streaming multiplies the representations of a given video stream, therefore diminishing the effectiveness of caching: namely, to get a hit for a chunk in the cache, it has to be for the same format and encoding values. Alternatively, to get the same hit rate as for a stream using a single encoding, the cache size must be scaled up to include all the possible representations.

- Caching introduces oscillatory dynamics as it may modify the estimation of the available bandwidth between the end user and the repository where it is getting the chunks from. For instance, if an edge cache holds a low resolution representation near the user, the user getting this low resolution chunks will observe a good performance, and will then request higher resolution chunks. If those are hosted on a server with poor performance, then the client would have to switch back to the low representation. This oscillation may be detrimental to the perceived QoE of the user.

- The ICN transport mechanism needs to be compatible to some extent with DASH. To take a CCN example, the rate at which interests are issued should be such that the chunks received in return arrive fast enough and with the proper encoding to keep the playback buffer above some threshold.

- The usage of multiple network interfaces is possible in ICN, enabling a seamless handover between them. For the combination with DASH, an intelligent strategy which should focus on traffic load balancing between the available links may be necessary. This would increase the effective media throughput of DASH by leveraging the combined available bandwidth of all links, however, it could potentially lead to high variations of the media throughput.

- DASH does not define how the MPD is retrieved; hence, this is compatible with CCN. However, the current profiles defined within MPEG-DASH require the MPD to contain HTTP-URLs (incl. http and https URI schemes) to identify segments. To enable a more integrated approach as described in this document, an additional profile for DASH over CCN has to be defined, enabling ICN/CCN-based URIs to identify and request the media segments.

We describe in Section 5.4 a potential implementation of a dynamic adaptive video stream over ICN, based upon DASH and CCN [5].

5.3.2. Interaction of ICN with Layered Encoding

Issues of interest to an Information-Centric network architecture in the context of layered video streaming include:

. Caching of the multiple layers. The caching priority should go to the base layer, and defining caching policy to decide when to cache enhancement layers;
. Synchronization of multiple content streams, as the multiple layers may come from different sources in the network (for instance, the base layer might be cached locally while the enhancement layers may be stored in the origin server). Video and audio video streams must be synchronized, and this includes both intra-layer synchronization (for the layers of the same video or audio stream) and inter-stream synchronization (see Section 9 for other synchronization aspects to be included in the "Future Steps for Video in ICN");
. Naming of the different layers: when the client requests an object, the request can be satisfied with the base layer alone, aggregated with enhancement layers. Should one request be sufficient to provide different streams? In a CCN architecture for instance, this would violate a one interest-one data packet principle and the client would need to specify each layer it would like to receive. In a Pub/Sub architecture, the rendezvous point would have to make a decision as to which layers (or which pointer to which layer’s location) to return.

5.4. Possible Integration of Video streaming and ICN architecture

5.4.1. DASH over CCN

DASH is intended to enable adaptive streaming, i.e., each content piece can be provided in different qualities, formats, languages, etc., to cope with the diversity of todays' networks and devices. As this is an important requirement for Future Internet proposals like CCN, the combination of those two technologies seems to be obvious. Since those two proposals are located at different protocol layers - DASH at the application and CCN at the network layer - they can be combined very efficiently to leverage the advantages of both and potentially eliminate existing disadvantages. As CCN is not based on classical host-to-host connections, it is possible to consume content from different origin nodes as well as over different network links in parallel, which can be seen as an intrinsic error resilience feature w.r.t. the network. This is a useful feature of CCN for adaptive multimedia streaming within mobile environments since most mobile devices are equipped with multiple network links like 3G and WiFi. CCN offers this functionality out of the box which is beneficial when used for DASH-based services. In particular, it is possible to enable adaptive video streaming handling both bandwidth and network link changes. That is, CCN handles the network link decision and DASH is implemented on top of CCN to adapt the video stream to the available bandwidth.

In principle, there are two options to integrate DASH and CCN: a proxy service acting as a broker between HTTP and CCN as proposed in [6], and the DASH client implementing a native CCN interface. The former transforms an HTTP request to a corresponding interest packet as well as a data packet back to an HTTP response, including reliable transport as offered by TCP. This may be a good compromise to implement CCN in a managed network and to support legacy devices. Since such a proxy is already described in [6] this draft focuses on a more integrated approach, aiming at fully exploiting the potential of a CCN DASH Client. That is, we describe a native CCN interface within the DASH client, which adopts a CCN naming scheme (CCN URIs) to denote segments in the Media Presentation Description (MPD). In this architecture, only the network access component on the client has to be modified and the segment URIs within MPD have to be updated according to the CCN naming scheme.

Initially, the DASH client retrieves the MPD containing the CCN URIs of the content representations including the media segments. The naming scheme of the segments may reflect intrinsic features of CCN like versioning and segmentation support. Such segmentation support is already compulsory for multimedia streaming in CCN and, thus, can

also be leveraged for DASH-based streaming over CCN. The CCN versioning can be adopted in a further step to signal different representations of the DASH-based content, which enables an implicit adaptation of the requested content to the clients' bandwidth conditions. That is, the interest packet already provides the desired characteristics of a segment (such as bit rate, resolution, etc.) within the content name (or potentially within parameters defined as extra types in the packet formats). Additionally, if bandwidth conditions of the corresponding interfaces or routing paths allow so, DASH media segments could be aggregated automatically by the CCN nodes, which reduces the amount of interest packets needed to request the content. However, such approaches need further research, specifically in terms of additional intelligence and processing power needed at the CCN nodes.

After requesting the MPD, the DASH client will start to request particular segments. Therefore, CCN interest packets are generated by the CCN access component and forwarded to the available interfaces. Within the CCN, these interest packets leverage the efficient interest aggregation for, e.g., popular content, as well as the implicit multicast support. Finally, the interest packets are satisfied by the corresponding data packets containing the video segment data, which are stored on the origin server or any CCN node, respectively. With an increasing popularity of the content, it will be distributed across the network resulting in lower transmission delays and reduced bandwidth requirements for origin servers and content providers respectively.

With the extensive usage of in-network caching, new drawbacks are introduced since the streaming logic is located at the client, i.e., clients are not aware of each other and the network infrastructure and cache states. Furthermore, negative effects are introduced when multiple clients are competing for a bottleneck and when caching is influencing this bandwidth competition. As mentioned above, the clients request individual portions of the content based on available bandwidth, which is calculated using throughput estimations. This uncontrolled distribution of the content influences the adaptation process of adaptive streaming clients. The impact of this falsified throughput estimation could be tremendous and leads to a wrong adaptation decision which may impact the Quality of Experience (QoE) at the client, as shown in [8]. In ICN, the client does not have the knowledge from which source the requested content is actually served or how many origin servers of the content are available, as this is transparent and depends on the name-based routing. This introduces the challenge that the adaptation logic of the adaptive streaming client is not aware of the event when the ICN routing decides to switch to a different

origin server or content is coming through a different link/interface. As most algorithms implementing the adaption logic are using bandwidth measurements and related heuristics, the adaptation decisions are no longer valid when changing origin servers (or links) and potentially cause playback interruptions and, consequently, stalling. Additionally, ICN supports the usage of multiple interfaces. A seamless handover between these interfaces (and different sources for the content) comes together with changes in performance, e.g., due to switching between fixed and wireless, 3G/4G and WiFi networks, or between different types of servers (say with/without SSD, or with/without hardware acceleration), etc.

Considering these characteristics of ICN, adaptation algorithms merely based on bandwidth measurements are not appropriate anymore, as potentially each segment can be transferred from another ICN node or interface, all with different bandwidth conditions. Thus, adaptation algorithms taking into account these intrinsic characteristics of ICN are preferred over algorithms based on mere bandwidth measurements.

5.4.2. Testbed, Open Source Tools, and Dataset

For the evaluations of DASH over CCN, a testbed with open source tools and datasets is provided in [9]. In particular, it provides two client player implementations, (i) a libdash extension for DASH over CCN and (ii) a VLC plugin implementing DASH over CCN. For both implementations the CCNx implementation has been used as a basis.

The general architecture of libdash is organized in modules, so that the library implements a MPD parser and an extensible connection manager. The library provides object-oriented interfaces for these modules to access the MPD and the downloadable segments. These components are extended to support DASH over CCN and available in a separate development branch of the github project available at http://www.github.com/bitmovin/libdash. libdash comes together with a fully featured DASH player with a QT-based frontend, demonstrating the usage of libdash and providing a scientific evaluation platform. As an alternative, patches for the DASH plugin of the VLC player are provided. These patches can be applied to the latest source code checkout of VLC resulting in a DASH over CCN-enabled VLC player.

Finally, a DASH over CCN dataset is provided in form of a CCNx repository. It includes 15 different quality representation of the well-known Big Buck Bunny Movie, ranging from 100 kbps up to 4500 kbps. The content is split into segments of two seconds, and described by an associated MPD using the presented naming scheme in Section 4.1. This repository can be downloaded from [9], and is also

provided by a public accessible CCNx node. Associated routing commands for the CCNx namespaces of the content are provided via scripts coming together with the dataset and can be used as a public testbed.

6. P2P video distribution and ICN

Another form of distributing content - and video in particular- which ICNs need to support is Peer-to-Peer distribution (P2P). We see now how an existing protocol such as PPSP can be modified to work in an ICN environment.

6.1. Introduction to PPSP

P2P video Streaming (PPS) is a popular approach to redistribute live media over Internet. The proposed P2PVS solutions can be roughly classified in two classes:

- Push/Tree based

- Pull/Mesh based

The Push/Tree based solution creates an overlay network among peers that has a tree shape [30]. Using a progressive encoding (e.g. Multiple Description Coding or H.264 Scalable Video Coding), multiple trees could be set up to support video rate adaptation. On each tree an enhancement stream is sent. The higher the number of received streams, the higher the video quality. A peer controls the video rate by fetching or not the streams delivered over the distribution trees.

The Pull/Mesh based solution is inspired by the BitTorrent file sharing mechanism. A Tracker collects information about the state of the swarm (i.e. set of participating peers). A peer forms a mesh overlay network with a subset of peers, and exchange data with them. A peer announces what data items it disposes and requests missing data items that are announced by connected peers. In case of live streaming, the involved data set includes only a recent window of data items published by the source. Also in this case, the use of a progressive encoding can be exploited for video rate adaptation.

Pull/Mesh based P2PVS solutions are the more promising candidate for the ICN deployment, since most of ICN approach provides a pull-based API [5][10][11][12]. In addition, Pull/Mesh based P2PVS are more robust than Push/Tree based one [13] and the Peer to Peer Streaming Protocol (PPSP) working group [14] is also proposing a Pull/Mesh based solution.

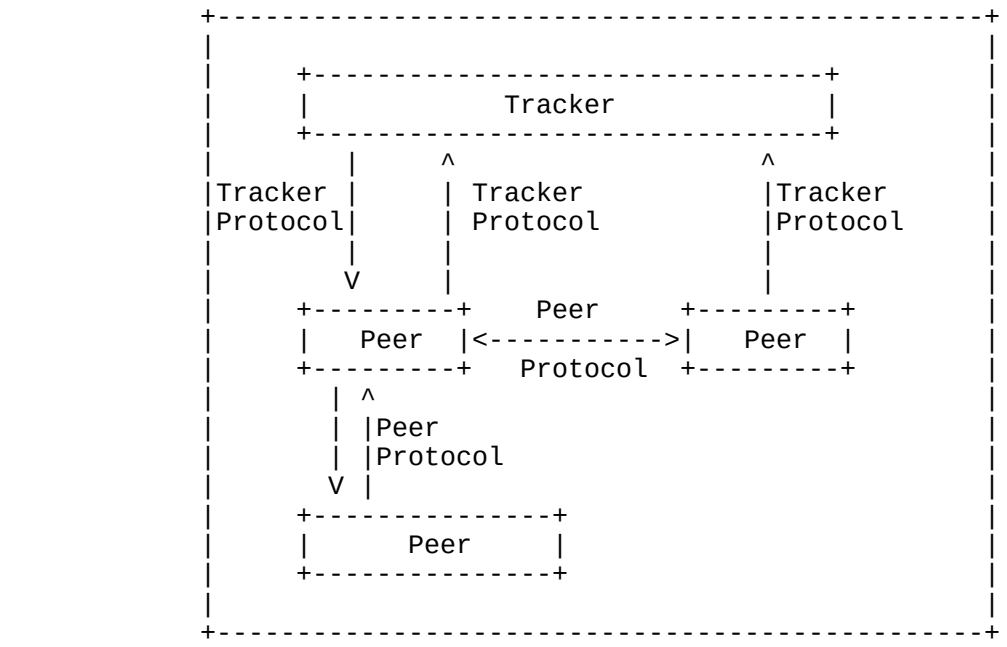


Figure 1: PPSP System Architecture (source [RFC6972])

Figure 1 reports the PPSP architecture presented in [RFC6972]. PEERs announce and share video chunks and a TRACKER maintains a list of PEERs participating in a specific audio/video channel or in the distribution of a streaming file. The tracker functionality may be centralized in a server or distributed over the PEERs. PPSP standardize the Peer and Tracker Protocols, which can run directly over UDP or TCP.

This document discusses some preliminary concepts about the deployment of PPSP on top of an ICN that exposes a pull-based API, meanwhile considering the impact of MPEG DASH streaming format.

6.2. PPSP over ICN: deployment concepts

6.2.1. PPSP short background

PPSP specifies peer protocol (PPSPP) [15] and tracker protocol (PPSP-TP)[16].

Some of the operations carried out by the tracker protocol are the followings. When a peer wishes to join the streaming session it contacts the Tracker (CONNECT message), obtains a PEER_ID and a list of PEER_IDs (and IP addresses) of other peers that are participating to the SWARM and that the tracker has singled out for the requesting peer (this may be a subset of the all peers of the SWARM). In addition to this join operation, a peer may contact the tracker to request to renew the list of participating peers (FIND message), to periodically update its status to the tracker (STAT_REPORT message), etc.

Some of the operations carried out by the peer protocol are the following. Using the list of peers delivered by the tracker, a peer establishes a session with them (HANDSHAKE message). A peer periodically announces to neighboring peers which chunks it has available for download (HAVE message). Using these announcements, a peer requests missing chunks from neighboring peers (REQUEST messages), which will send back them (DATA message).

6.2.2. From PPSP messages to ICN named-data

An ICN provides users with data items exposed by names. The bundle name and data item is usually referred as named-data, named-content, etc. To transfer PPSP messages though an ICN the messages should be wrapped as named-data items, and receivers should request them by name.

A PPSP entity receives messages from peers and/or tracker. Some operations require gathering the messages generated by another specific host (peer or tracker). For instance, if a peer A wishes to gain information about video chunks available from peer B, the former shall fetch the PPSP HAVE messages specifically generated by the later. We refer to these kinds of named-data as "located-named-data", since they should be gathered from a specific location (e.g. peer B).

For other PPSP operations, such as fetching a DATA message (i.e. a video chunk), as long as a peer receives the requested content, it doesn’t matter which endpoint generated the data.We refer to this information with the generic term "named-data".

The naming scheme differentiates named-data and located-named-data items. In case of named-data, the naming scheme only includes a content identifier (e.g. the name of the video chunk), without any prefix identifying who provides the content. For instance, a DATA message containing the video chunk #1 may be named as "ccnx:/swarmID/chunk/chunkID", where swarmID is a unique identifier

of the streaming session, "chunk" is a keyword and chunkID is the chunk identifier (e.g. a integer number).

In case of located-named-data, the naming scheme includes a location-prefix, which uniquely identifies the host generating the data item. This prefix may be the PEER_ID in case the host was a peer or a tracker identifier in case the host was the tracker. For instance, a HAVE message generated by a peer B may be named as "ccnx:/swarmID/peer/PEER_ID/HAVE", where "peer" is a keyword, PEER_ID_B is the identifier of peer B and HAVE is a keyword.

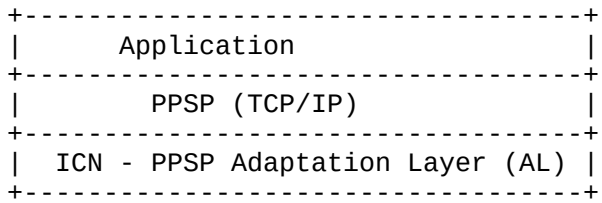
6.2.3. Support of PPSP interaction through a pull-based ICN API

The PPSP procedures are based both on pull and push interactions. For instance, the distribution of chunks availability can be classified as a push-based operation, since a peer sends an "unsolicited" information (HAVE message) to neighboring peers. Conversely the procedure used to receive video chunks can be classified as pull-based, since it is supported by a request/response interaction (i.e. REQUEST, DATA messages).

As we said, we refer to an ICN architecture which provides a pull-based API. Accordingly, the mapping of PPSP pull-based procedure is quite simple. For instance, using the CCN architecture [5] a PPSP DATA message may be carried by a CCN Data message and a REQUEST message can transferred by a CCN Interest.

Conversely, the support of push-based PPSP operations may be more difficult. We need of an adaptation functionality that carries out a push-based operation using the underlying pull-based service primitives. For instance, a possible approach is to use the request/response (i.e. Interest/Data) four ways handshakes proposed in [7]. Another possibility is that receivers periodically send out request messages of the named-data that neighbors will push and, when available, sender inserts the pushed data within a response message.

6.2.4. Abstract layering for PPSP over ICN



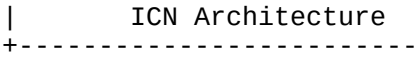


Figure 2: Mediator approach

Figure 2 provides a possible abstract layering for PPSP over ICN. The Adaptation Layer acts as a mediator (proxy) between legacy PPSP entities based on TCP/IP and the ICN architecture. In facts, the role the mediator is to use ICN to transfer PPSP legacy messages.

This approach makes possible to merely reuse TCP/IP P2P applications whose software includes also PPSP functionality. This "all-in-one" development approach may be rather common since the PPSP-Application interface is not going to be specified. Moreover, if the Operating System will provide libraries that expose a PPSP API, these will be initially based on a underlying TCP/IP API. Also in this case, the mediator approach would make possible to easily reuse both the PPSP libraries and the Application on top of an ICN.

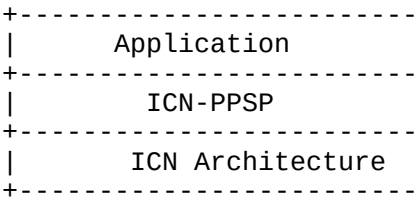


Figure 3: Clean-slate approach

Figure 3 sketches a clean-slate layering approach in which the application directly includes or interacts with a PPSP version based on ICN. Likely such a PPSP_ICN integration could yield a simplier development, also because it does not require implementing a TCP/IP to ICN translation as in the Mediator approach. However, the clean-slate approach requires developing the application (in case of embedded PPSP functionality) or the PPSP library from scratch, without exploiting what might already exist for TCP/IP.

Overall, the Mediator approach may be considered as the first step of a migration path towards ICN native PPSP applications.

6.2.5. PPSP interaction with the ICN routing plane

Upon the ICN API a user (peer) requests a content and the ICN sends it back. The content is gathered by the ICN from any source, which could be the closest peer that disposes of the named-data item, an in-network cache, etc. Actually, "where" to gather the content is

controlled by an underlying ICN routing plane, which sets up the ICN forwarding tables (e.g. CCN FIB [5]).

A cross-layer interaction between the ICN routing plane and the PPSP may be required to support a PPSP session. Indeed, ICN shall forward request messages (e.g. CCN Interest) towards the proper peer that can handle them. Depending on the layering approach, this cross-layer interaction is controlled either by the Adaptation Layer or by the ICN-PPSP. For example, if a peer A receives a HAVE message indicating that peer B disposes of the video chunk named "ccnx:/swarmID/chunk/chunkID", then former should insert in its ICN forwarding table an entry for the prefix "ccnx:/swarmID/chunk/chunkID" whose next hop locator (e.g. IP address) is the network address of peer B [17].

6.2.6. ICN deployment for PPSP

The ICN functionality that supports a PPSP session may be "isolated" or "integrated" with the one of a public ICN.

In the isolated case, a PPSP session is supported by an instance of an ICN (e.g. deployed on top of IP), whose functionalities operate only on the limited set of nodes participating to the swarm, i.e. peers and the tracker. This approach resembles the one followed by current P2P application, which usually form an overlay network among peers of a P2P application. And intermediate public IP routers do not carry out P2P functionalities.

In the integrated case, the nodes of a public ICN may be involved in the forwarding and in-network caching procedures. In doing so, the swarm may benefit from the presence of in-network caches so limiting uplink traffic on peers and inter-domain traffic too. These are distinctive advantages of using PPSP over a public ICN, rather than over TCP/IP. In addition, such advantages aren’t likely manifested in the case of isolated deployment.

However, the possible interaction between the PPSP and the routing layer of a public ICN may be dramatic, both in terms of explosion of the forwarding tables and in terms of security. These issues specifically take place for those ICN architectures for which the name resolution (i.e. name to next-hop) occurs en-route, like the CCN architecture.

For instance, using the CCN architecture, to fetch a named-data item offered by a peer A the on-path public ICN entities have to route the request messages towards the peer A. This implies that the ICN forwarding tables of public ICN nodes may contain many entries, e.g.

one entry per video chunk, and these entries are difficult to be aggregated since peers may have available only sparse parts of a big content, whose names have a same prefix (e.g. "ccnx:/swarmID"). Another possibility is to wrap all PPSP messages into a located-named-data. In this case the forwarding tables should contain "only" the PEER_ID prefixes (e.g. "ccnx:/swarmID/peer/PEER_ID"), so scaling down the number of entries from number of chunks to number of peers. However, in this case the ICN mechanisms recognize a same video chunk offered by different peers as different contents, so vanishing caching and multicasting ICN benefits. Moreover, in any case routing entries should be updated either the base of the availability of named-data items on peers or on the presence of peers, and these events in a P2P session is rapidly changing so possibly hampering the convergence of the routing plane. Finally, since peers have an impact on the ICN forwarding table of public nodes, this may open obvious security issues.

6.3. Impact of MPEG DASH coding schemes

The introduction of video rate adaptation may significantly decrease the effectiveness of P2P cooperation and of in-network caching, depending of the kind of the video coding used by the MPEG DASH stream.

In case of a MPEG DASH streaming with MPEG AVC encoding, a same video chunk is independently encoded at different rates and the encoding output is a different file for each rate. For instance, in case of a video encoded at three different rates R1,R2,R3, for each segment S we have three distinct files: S.R1, S.R2, S.R3. These files are independent of each other. To fetch a segment coded at R2 kbps, a peer shall request the specific file S.R2. Receiver-driven algorithms, implemented by the video client, usually handle the estimation of the best coding rate.

The independence among files associated to different encoding rates and the heterogeneity of peer bandwidths, may dramatically reduce the interaction among peers, the effectiveness of in-network caching (in case of integrated deployment), and consequently the ability of PPSP to offload the video server (i.e. a seeder peer). Indeed, a peer A may select a coding rate (e.g. R1) different from the one selected by a peer B (e.g. R2) and this prevents the former to fetch video chunks from the later, since peer B only has chunks available that are coded at a rate different from the ones needed by A. To overcome this issue, a common distributed rate selection algorithm could force peers to select the same coding rate [17]; nevertheless this approach may be not feasible in the in case of many peers.

The use of SVC encoding (Annex G extension of the H.264/MPEG-4 AVC video compression standard) should make rate adaptation possible, meanwhile neither reducing peer collaborations nor the in-network caching effectiveness. For a single video chunk, a SVC encoder produces different files for the different rates (roughly "layers"), and these files are progressively related each other. Starting from a base-layer which provides the minimum rate encoding, the next rates are encoded as an "enhancement layer" of the previous one. For instance, in case the video is coded with three rates R1 (base-layer), R2 (enhancement-layer n.1), R3 (enhancement-layer n.2), then for each DASH segment we have three files S.R1, S.R2 and S.R3. The file S.R1 is the segment coded at the minimum rate (base-layer). The file S.R2 enhances S.R1, so as S.R1 and S.R2 can be combined to obtain a segment coded at rate R2. To get a segment coded at rate R2, a peer shall fetch both S.R1 and S.R2. This progressive dependence among files that encode a same segment at different rates makes peer cooperation possible, also in case peers player have autonomously selected different coding rates. For instance, if peer A has selected the rate R1, the downloaded files S.R1 are useful also for a peer B that has selected the rate R2, and vice versa.

7. IPTV and ICN

7.1. IPTV challenges

IPTV refers to the delivery of quality content broadcast over the Internet, and is typically associated with strict quality requirements, i.e., with a perceived latency of less than 500 ms and a packet loss rate that is multiple orders lower than the current loss rates experienced in the most commonly used access networks (see [31]). We can summarize the major challenges for the delivery of IPTV service as follows.

Channel change latency represents a major concern for the IPTV service. Perceived latency during channel change should be less than 500ms. To achieve this objective over the IP infrastructure, we have multiple choices:

(i) receiving fast unicast streams from a dedicated server (most effective but not resource efficient);
(ii) connecting to other peers in the network (efficiency depends on peer support, effective and resource efficient, if also supported with a dedicated server);

(iii) connecting to multiple multicast sessions at once (effective but not resource efficient, and depends on the accuracy of the prediction model used to track user activity).

The second major challenge is the error recovery. Typical IPTV service requirements dictate the mean time between artifacts to be approximately 2 hours (see [31]). This suggests the perceived loss rate to be around or less than 10^-7. Current IP-based solutions rely on the following proactive and reactive recovery techniques: (i) joining the FEC multicast stream corresponding to the perceived packet loss rate (not efficient as the recovery strength is chosen based on worst-case loss scenarios), (ii) making unicast recovery requests to dedicated servers (requires active support from the service provider), (iii) probing peers to acquire repair packets (finding matching peers and enabling their cooperation is another challenge).

7.2. ICN benefits for IPTV delivery

ICN presents significant advantages for the delivery of IPTV traffic. For instance, ICN inherently supports multicast and allows for quick recovery from packet losses (with the help of in-network caching). Similarly, peer support is also provided in the shape of in-network caches that typically act as the middleman between two peers, enabling therefore earlier access to IPTV content.

However, despite these advantages, delivery of IPTV service over Information Centric Networks brings forth new challenges. We can list some of these challenges as follows:

. Messaging overhead: ICN is a pull-based architecture and relies on a unique balance between requests and responses. A user needs to make a request for each data packet. In the case of IPTV, with rates up to, and likely to be, above 15Mbps, we observe significant traffic upstream to bring those streams. As the number of streams increases (including the same session at different quality levels and other formats), so does the burden on the routers. Even if the majority of requests are aggregated at the core, routers close to the edge (where we observe the biggest divergence in user requests) will experience a significant increase in overhead to process these requests. The same is true at the user side, as the uplink usage multiplies

in the number of sessions a user requests (for instance, to minimize the impact of bandwidth fluctuations).

. Cache control: As the IPTV content expires at a rapid rate (with a likely expiry threshold of 1s), we need solutions to effectively flush out such content to also prevent degradation impact on other cached content, with the help of intelligently chosen naming conventions. However, to allow for fast recovery and optimize access time to sessions (from current or new users), the timing of such expirations needs to be adaptive to network load and user demand. However, we also need to support quick access to earlier content, whenever needed, for instance, when the user accesses the rewind feature (note that in-network caches will not be of significant help in such scenarios due to overhead required to maintain such content).

. Access accuracy: To receive the up-to-date session data, users need to be aware of such information at the time of their request. Unlike IP multicast, since the users join a session indirectly, session information is critical to minimize buffering delays and reduce the startup latency. Without such information, and without any active cooperation from the intermediate routers, stale data can seriously undermine the efficiency of content delivery. Furthermore, finding a cache does not necessarily equate to joining a session, as the look-ahead latency for the initial content access point may have a shorter lifetime than originally intended. For instance, if the user that has initiated the indirect multicast leaves the session early, the requests from the remaining users need to experience an additional latency of one RTT as they travel towards the content source. If the startup latency is chosen depending on the closeness to the intermediate router, going to the content source in-session can lead to undesired pauses.

It should be noted that IPTV includes more than just multicast. Many implementations include "trick plays" (fast forward, pause, rewind) that often transform a multicast session into multiple unicast sessions. In this context, ICN is beneficial, as the caching offers an implicit multicast, but without tight synchronization constraints in between two different users. One user may rewind, and start playing forward again, drawing from a nearby cache of the content recently viewed by another user (whereas in a strict multicast session, the opportunity of one user lagging off behind would be more difficult to implement).

8. Digital Rights Managements in ICN

This section discusses the need for Digital Rights Management (DRM) functionalities for multimedia streaming over ICN. It focuses on two possible approaches: modifying AAA to support DRM in ICN, and using Broadcast Encryption.

It is assumed that ICN will be used heavily for digital content dissemination. It is vital to consider DRM for digital content distribution. In today's Internet there are two predominant classes of business models for on-demand video streaming. The first model is based on advertising revenues. Non-copyright protected (usually user-generated content, UGC) is offered by large infrastructure providers like Google (YouTube) at no charge. The infrastructure is financed by spliced advertisements into the content. In this context DRM considerations may not be required, since producers of UGC may only strive for the maximum possible dissemination. Some producers of UGC are mainly interested to share content with their families, friends, colleges or others and have no intention to make profit. However, the second class of business models requires DRM, because they are primarily profit oriented. For example, large on-demand streaming platforms like Netflix establish business models based on subscriptions. Consumers may have to pay a monthly fee in order to get access to copyright protected content like TV series, movies or music. This model may be ad-supported and free to the content consumer, like YouTube Channels or Spotify. But the creator of the content expects some remuneration for his work. From the perspective of the service providers and the copyright owners, only clients that pay the fee (explicitly or implicitly through ad placement) should be able to access and consume the content. Anyway, the challenge is to find an efficient and scalable way of access control to digital content, which is distributed in information-centric networks.

8.1. Broadcast Encryption for DRM in ICN

The section discusses Broadcast Encryption (BE) as a suitable basis for DRM functionalities in conformance to the ICN communication paradigm. Especially when network inherent caching is considered the advantage of BE will be highlighted.

In ICN, data packets can be cached inherently in the network and any network participant can request a copy of these packets. This makes it very difficult to implement an access control for content that is distributed via ICN. A naive approach is to encrypt the transmitted data for each consumer with a distinct key. This prohibits everyone

other than the intended consumers to decrypt and consume the data. However, this approach is not suitable for ICN's communication paradigm since it would reduce the benefits gained from the inherent network caching. Even if multiple consumers request the same content the requested data for each consumer would differ using this approach. A better but still insufficient idea is to use a single key for all consumers. This does not destruct the benefits of ICN's caching ability. The drawback is that if one of the consumers illegally distributes the key, the system is broken and any entity in the network can access the data. Changing the key after such an event is useless since the provider has no possibility to identify the illegal distributer. Therefore this person cannot be stopped from distributing the new key again. In addition to this issue other challenges have to be considered. Subscriptions expire after a certain time and then it has to be ensured that these consumers cannot access the content anymore. For a provider that serves millions of daily consumers (e.g. Netflix) there could be a significant number of expiring subscriptions per day. Publishing a new key every time a subscription expires would require an unsuitable amount of computational power just to re-encrypt the collection of audio-visual content.

A possible approach to solve these challenges is Broadcast Encryption (BE) [22] as proposed in [23]. From this point on, this section will focus only on BE as an enabler for DRM functionality in the use case of ICN video streaming. This subsection continues with the explanation of how BE works and shows how BE can be used to implement an access control scheme in the context of content distribution in ICN.

BE actually carries a misleading name. One might expect a concrete encryption scheme. However, it belongs to the family of key-management schemes (KMS). KMS are responsible for the generation, exchange, storage and replacement of cryptographic keys. The most interesting characteristics of Broadcast Encryption Schemes (BES) are:

- A BES typically uses a global trusted entity called the licensing agent (LA), which is responsible for spreading a set of pre-generated secrets among all participants. Each participant gets a distinct subset of secrets assigned from the LA.
- The participants can agree on a common session key, which is chosen by the LA. The LA broadcasts an encrypted message that includes the key. Participants with a valid set of secrets can derive the session-key from this message.

- The number of participants in the system can change dynamically. Entities may join or leave the communication group at any time. If a new entity joins the LA passes on a valid set of secrets to that entity. If an entity leaves (or is forced to leave) the LA revokes the entity's subset of keys, which means that it cannot derive the correct session key anymore when the LA distributes a new key.
- Traitors (entities that reveal their secrets) can be traced and excluded from ongoing communication. The algorithms and preconditions to identify a traitor vary between concrete BES.

This listing already illustrates why BE is suitable to control the access to data that is distributed via an information-centric network. BE enables the usage of a single session key for confidential data transmission between a dynamically changing subset or network participants. ICN caches can be utilized since the data is encrypted only with a single key known by all legitimate clients. Furthermore, traitors can be identified and removed from the system. The issue of re-encryption still exists, because the LA will eventually update the session key when a participant should be excluded. However, this disadvantage can be relaxed in some way if the following points are considered:

- The updates of the session key can be delayed until a set of compromised secrets has been gathered. Note that secrets may become compromised because of two reasons. First, a traitor could have illegally revealed the secret. Second, the subscription of an entity expired. Delayed revocation temporarily enables some non-legitimate entities to consume content. However, this should not be a severe problem in home entertainment scenarios. Updating the session key in regular (not too short) intervals is a good tradeoff. The longer the interval last the less computational resources are required for content re-encryption and the better the cache utilization in the ICN will be. To evict old data from ICN caches that has been encrypted with the prior session key the publisher could indicate a lifetime for transmitted packets.
- Content should be re-encrypted dynamically at request time. This has the benefit that untapped content is not re-encrypted if the content is not requested during two session key updates and therefore no resources are wasted. Furthermore, if the updates are triggered in non-peak times the maximum amount of resource needed at one point in time can be lowered effectively, since in peak times generally more diverse content is requested.
- Since the amount of required computational resources may vary strongly from time to time it would be beneficial for any

streaming provider to use cloud-based services to be able to dynamically adapt the required resources to the current needs. Regarding to a lack of computation time or bandwidth the cloud service could be used to scale up to overcome shortages.

Figure 4 show the potential usage of BE in a multimedia delivery frameworks that builds upon ICN infrastructure and uses the concept of dynamic adaptive streaming, e.g., DASH. BE would be implemented on the top to have an efficient and scalable way of access control to the multimedia content.

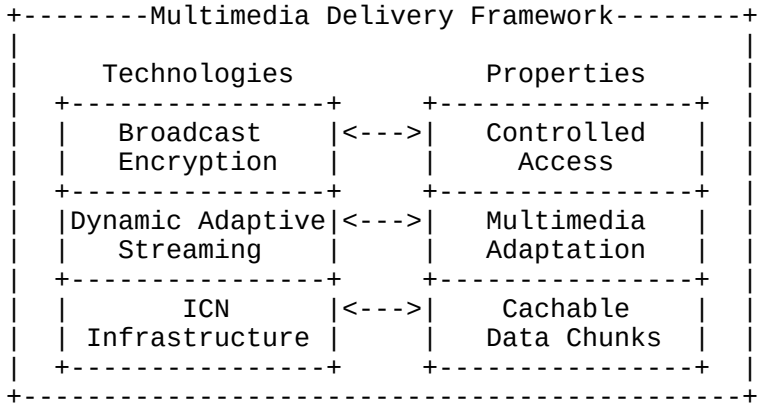


Figure 4: A potential multimedia framework using BE.

8.2 . AAA Based DRM for ICN Networks

8.2.1. Overview

Recently, a novel approach to Digital Rights Management (DRM) has emerged to link DRM to usual network management operations, hence linking DRM to authentication, authorization, and accounting (AAA) services. ICN provides the abstraction of an architecture where content is requested by name and could be served from anywhere. In DRM, the content provider (the origin of the content) allows the destination (the end user account) to use the content. The content provider and content storage/cache are at two different entities in ICC and for traditional DRM only source and destination count and not the intermediate storage. The proposed solution allows the provider of the caching to be involved in the DRM policies using well known AAA mechanisms. It is important to note that this solution is compatible with the proposes the Broadcast Encryption

(BE) proposed earlier in this draft. The BE proposes a technology as this solution is more operational.

8.2.2. Implementation

With the proposed AAA-based DRM, when content is requested by name from a specific destination, the request could link back to both the content provider and the caching provider via traditional AAA mechanisms, and trigger the appropriate DRM policy independently from where the content is stored. In this approach the caching, DRM and AAA remain independent entities but can work together through ICN mechanisms. The proposed solution enables extending the traditional DRM done by the content provider to jointly being done by content provider and network/caching provider.

The solution is based on the concept of a "token". The content provider authenticates the end user and issues an encrypted token to authenticate the a named content ID or IDs that the user can access. The token will be shared with the network provider and used as the interface to the AAA protocols. At this point all content access is under the control of the network provider and the ICN. The controllers and switches can manage the content requests and handle mobility. The content can be accessed from anywhere as long as the token remains valid or the content is available in the network. In such a scheme the content provider does not need to be contacted every time a named content is requested. This reduces the load of the content provider network and creates a DRM mechanism that is much more appropriate for the distributed caching and peer-to-peer storage characteristic of ICN networks. In particular, the content requested by name can be served from anywhere under the only condition that the storage/cache can verify that the token is valid for content access.

The solution is also fully customizable to both content and network provider’s needs as the tokens can be issued based on user accounts, location and hardware (MAC address for example) linking it naturally to legacy authentication mechanisms. In addition, since both content and network providers are involved in DRM policies pollution attacks and other illegal requests for the content can be more easily detected. The proposed AAA-based DRM is currently under full development.

9. Future Steps for Video in ICN

The explosion of online video services, along with their increased consumption by mobile wireless terminals, further exacerbates the challenges of Video Adaptation leveraging ICN mechanisms. The following sections present a series of research items derived from these challenges, further introducing next steps for the subject.

9.1. Large Scale Live Events

An active area of investigation and a potential use case where ICN would provide significant benefits, is that of distributing content, and video in particular, using local communications in large scale events such as sports event in a stadium, a concert or a large demonstration.

Such use-case involves locating content that is generated on the fly and requires discovery mechanisms in addition to sharing mechanisms. The scalability of the distribution becomes important as well.

9.2. Video Conferencing and Real-Time Communications

Current protocols for video-conferencing have been designed, and this document needs to take input from them to identify the key research issues. Real-time communications add timing constraints (both in terms of delay and in terms of synchronization) to the scenario discussed above.

AR and VR (and immersive multimedia experiences in general) are clearly an area of further investigation, as they involve combining multiple streams of data from multiple users into a coherent whole. This raises issues of multi-source multi-destination multimedia streams that ICN may be equipped to deal with in a more natural manner than IP that is inherently unicast.

9.3. Store-and-Forward Optimized Rate Adaptation

One of the benefits of ICN is to allow the network to insert caching in the middle of the data transfer. This can be used to reduce the overall bandwidth demands over the network by caching content for future re-use. But it provides more opportunities for optimizing video streams.

Consider for instance the following scenario: a client is connected via an ICN network to a server. Let’s say the client is connected wirelessly to a node that has a caching capability, which is connected through a WAN to the server. Assume further that the

capacity of each of the links (both the wireless and the WAN logical links) vary with time.

If the rate adaptation is provided in an end-to-end manner, as in current mechanisms like DASH, then the maximal rate that can be supported at the client is that of the minimal bandwidth on each link.

For instance, if during time period 1, the wireless capacity is 1 and the wired capacity is 2, and during time period 2, the wireless is 2 due to some hotspot, and the wired is 1 due to some congestion in the network, then the best end-to-end rate that can be achieved is 1 during each period.

However, if the cache is used during time period 1 to pre-fetch 2 units of data, then during period 2, there is 1 unit of data at the cache, and another unit of data, which can be streamed from the server, and the rate that can be achieved is therefore 2 units of data. In this case, the average bandwidth rises from 1 to 1.5 over the 2 periods.

This straw man example illustrate a) the benefit of ICN for increasing the throughput of the network, and b) the need for the special rate adaptation mechanisms to be designed so as to take advantage of this gain. End-to-end rate adaptation cannot take advantage of the cache availability.

9.4. Heterogeneous Wireless Environment Dynamics

With the ever-growing increase in online services being accessed by mobile devices, operators have been deploying different overlapping wireless access networking technologies. In this way, in the same area, user terminals are within range of different cellular, Wi-Fi or even WiMAX networks. Moreover, with the advent of the Internet of Things (e.g., surveillance cameras feeding video footage), this list can be further complemented with more specific short-range technologies, such as Bluetooth or ZigBee.

In order to leverage from this plethora of connectivity opportunities, user terminals are coming equipped with different wireless access interfaces, providing them with extended connectivity opportunities. In this way, such devices become able to select the type of access which best suits them according to different criteria, such as available bandwidth, battery consumption, access do different link conditions according to the user profile or even access to different content. Ultimately, these aspects contribute to the Quality of Experience perceived by the

end-user, which is of utmost importance when it comes to video content.

However, the fact that these users are mobile and using wireless technologies, also provides a very dynamic setting, where the current optimal link conditions at a specific moment might not last or be maintained while the user moves. These aspects have been amply analyzed in recently finished projects such as FP7 MEDIEVAL [18], where link events reporting on wireless conditions and available alternative connection points were combined with vide requirements and traffic optimization mechanisms, towards the production of a joint network and mobile terminal mobility management decision. Concretely, in [19] link information about the deterioration of the wireless signal was sent towards a mobility management controller in the network. This input was combined with information about the user profile, as well as of the current video service requirements, and used to trigger the decrease or increase of scalable video layers, adjusting the video to the ongoing link conditions. Incrementally, the video could also be adjusted when a new better connectivity opportunity presents itself.

In this way, regarding Video Adaptation, ICN mechanisms can leverage from their intrinsic multiple source support capability and go beyond the monitoring of the status of the current link, thus exploiting the availability of different connectivity possibilities (e.g., different "interfaces"). Moreover, information obtained from the mobile terminal’s point of view of its network link, as well as information from the network itself (i.e., load, policies, and others), can generate scenarios where such information is combined in a joint optimization procedure allowing the content to be forward to users using the best available connectivity option (e.g., exploiting management capabilities supported by ICN intrinsic mechanisms as in [20]).

In fact, ICN base mechanisms can further be exploited in enabling new deployment scenarios such as preparing the network for mass requests from users attending a large multimedia event (i.e., concert, sports), allowing video to be adapted according to content, user and network requirements and operation capabilities in a dynamic way.

The enablement of such scenarios require further research, with the main points highlighted as follows:

. Development of a generic video services (and obviously content) interface allowing the definition and mapping of their

requirements (and characteristics) into the current capabilities of the network;

. How to define a scalable mechanism allowing either the video application at the terminal, or some kind of network management entity, to adapt the video content in a dynamic way;

. How to develop the previous research items using intrinsic ICN mechanisms (i.e., naming and strategy layers);

. Leverage intelligent pre-caching of content to prevent stalls and poor quality phases, which lead to bad Quality of Experience of the user. This includes in particular the usage in mobile environments, which are characterized by severe bandwidth changes as well as connection outages, as shown in [21];

. How to take advantage of the multi-path opportunities over the heterogeneous wireless interfaces.

9.5. Network Coding for Video Distribution in ICN

An interesting research area for combining heterogeneous sources is to use network coding [24]. Network coding allows to asynchronously combine multiple sources by having each of them send information that is not duplicated by the other but can be combined to retrieve the video stream.

However, this creates issues in ICN in terms of defining the proper rate adaptation for the video stream; securing the encoded data; caching the encoded data; timeliness of the encoded data; overhead of the network coding operations both in network resources and in added buffering delay, etc.

Network coding has shown promise in reducing buffering events in unicast, multicast and P2P setting. [26] considers strategies using network coding to enhance QoE for multimedia communications. Network coding can be applied to multiple streams, but also within a single stream as an equivalent of a composable erasure code. Clearly, there is a need for further investigation of network coding in ICN, potentially as a topic of activity in the research group.

9.6. Synchronization Issues for Video Distribution in ICN

ICN de-couples the fetching of video chunks from the location of these chunks. This means an audio chunk may be received from one network element (cache/storage/server) while a video chunk may be received from another one while another chunk (say, the next one, or

another layer from the same video stream) may come from a third element. This introduces disparity in the retrieval times and locations of the different elements of a video stream that need to be played at the same (or almost same) time. Synchronization of such delivery and playback may require specific synchronization tools for video delivery in ICN.

Other synchronization aspects involve:

- synchronizing within a single stream, for instance the consecutive chunks of a single stream, or the multiple layers of a layered scheme, when sources and transport layers may be different. Re-ordering the packets of a stream distributed over multiple sources at the video client, or ensuring that multiple chunks coming from multiple sources arrive within an acceptable time window;
- synchronizing multiple streams, such as the audio and video components of a video stream, which can be received from independent sources;
- synchronizing multiple streams from multiple sources to multiple destinations, such as mass distribution of live events. For instance, for live video streams or video-conferencing, some level of synchronization is required so that people watching the stream view the same events at the same time.

Some of these issues were addressed in [27] in the context of social video consumption. Network coding, with traffic engineering, is considered as a potential solution for synchronization issues. Other approaches could be considered that are specific for ICN as well.

Traffic engineering in ICN [28,29] may be required to provide proper synchronization of multiple streams.

10. Security Considerations

This is informational. There are no specific security considerations outside of those mentioned in the text.

11. IANA Considerations

This document does not require any IANA action.

12. Conclusions

This draft proposed adaptive video streaming for ICN, identified potential problems and presented the combination of CCN with DASH as a solution. As both concepts, DASH and CCN, maintain several elements in common, like, e.g., the content in different versions

being dealt with in segments, combination of both technologies seems useful. Thus, adaptive streaming over CCN can leverage advantages such as, e.g., efficient caching and intrinsic multicast support of CCN, routing based on named data URIs, intrinsic multi-link and multi-source support, etc.

In this context, the usage of CCN with DASH in mobile environments comes together with advantages compared to today's solutions, especially for devices equipped with multiple network interfaces. The retrieval of data over multiple links in parallel is a useful feature, specifically for adaptive multimedia streaming, since it offers the possibility to dynamically switch between the available links depending on their bandwidth capabilities, transparent to the actual DASH client.

13. References

13.1. Normative References

[RFC6972] Y. Zhang, N. Zong, "Problem Statement and Requirements of the Peer-to-Peer Streaming Protocol (PPSP)", RFC6972, July 2013

13.2. Informative References

[1] ISO/IEC DIS 23009-1.2, Information technology - Dynamic adaptive streaming over HTTP (DASH) - Part 1: Media presentation description and segment formats

[2] Lederer, S., Mueller, C., Rainer, B., Timmerer, C., Hellwagner, H., "An Experimental Analysis of Dynamic Adaptive Streaming over HTTP in Content Centric Networks", in Proceedings of the IEEE International Conference on Multimedia and Expo 2013, San Jose, USA, July, 2013

[3] Liu, Y., Geurts, J., Point, J., Lederer, S., Rainer, B., Mueller, C., Timmerer, C., Hellwagner, H., "Dynamic Adaptive Streaming over CCN: A Caching and Overhead Analysis", in Proceedings of the IEEE international Conference on Communication (ICC) 2013 - Next-Generation Networking Symposium, Budapest, Hungary, June, 2013

[4] Grandl, R., Su, K., Westphal, C., "On the Interaction of Adaptive Video Streaming with Content-Centric Networks", eprint arXiv:1307.0794, July 2013.

[5] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs and R. Braynard, "Networking named content", in Proc. of the 5th int. Conf. on Emerging Networking Experiments and Technologies (CoNEXT '09). ACM, New York, NY, USA, 2009, pp. 1-12.

[6] A. Detti, M. Pomposini, N. Blefari-Melazzi, S. Salsano and A. Bragagnini, "Offloading cellular networks with Information-Centric Networking: The case of video streaming", In Proc. of the Int. Symp. on a World of Wireless, Mobile and Multimedia Networks (WoWMoM '12), IEEE, San Francisco, CA, USA, 1-3, 2012.

[7] V. Jacobson, D. K. Smetters, N. H. Briggs, M. F. Plass, P. Stewart, J. D. Thornton, and R. L. Braynard, "VoCCN: Voice over content-centric networks," in ACM ReArch Workshop, 2009

[8] Christopher Mueller, Stefan Lederer and Christian Timmerer, A proxy effect analysis and fair adaptation algorithm for multiple competing dynamic adaptive streaming over HTTP clients, In Proceedings of the Conference on Visual Communications and Image Processing (VCIP) 2012, San Diego, USA, November 27-30, 2012.

[9] DASH Research at the Institute of Information Technology, Multimedia Communication Group, Alpen-Adria Universitaet Klagenfurt, URL: http://dash.itec.aau.at

[10] A. Detti, N. Blefari-Melazzi, S. Salsano, and M. Pomposini CONET: A content centric inter-networking architecture," in ACM Workshop on Information-Centric Networking (ICN), 2011.

[11] W. K. Chai, N. Wang, I. Psaras, G. Pavlou, C. Wang, G. C. de Blas, F. Ramon-Salguero, L. Liang, S. Spirou, A. Beben, and E. Hadjioannou, "CURLING: Content-ubiquitous resolution and delivery infrastructure for next-generation services," IEEE Communications Magazine, vol. 49, no. 3, pp. 112-120, March 2011

[12] NetInf project Website http://www.netinf.org

[13] N. Magharei, R. Rejaie, Yang Guo, "Mesh or Multiple-Tree: A Comparative Study of Live P2P Streaming Approaches," INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE , vol., no., pp.1424,1432, 6-12 May 2007

[14] PPSP WG Website https://datatracker.ietf.org/wg/ppsp/

[15] A. Bakker, R. Petrocco, V. Grishchenko, "Peer-to-Peer Streaming Peer Protocol (PPSPP)", draft-ietf-ppsp-peer-protocol-08

[16] Rui S. Cruz, Mario S. Nunes, Yingjie Gu, Jinwei Xia, Joao P. Taveira, Deng Lingli, "PPSP Tracker Protocol-Base Protocol (PPSP-TP/1.0)", draft-ietf-ppsp-base-tracker-protocol-02

[17] A.Detti, B. Ricci, N. Blefari-Melazzi,"Peer-To-Peer Live Adaptive Video Streaming for Information Centric Cellular Networks", IEEE PIMRC 2013,London, UK, 8-11 September 2013

[18] http://www.ict-medieval.eu

[19] B. Fu, G. Kunzmann, M. Wetterwald, D. Corujo, R. Costa, "QoE-aware Traffic Management for Mobile Video Delivery", Proc. 2013 IEEE ICC, Workshop on Immersive & Interactive Multimedia Communications over the Future Internet (IIMC), Budapest, Hungary, Jun 2013.

[20] Corujo D., Vidal I., Garcia-Reinoso J., Aguiar R., "A Named Data Networking Flexible Framework for Management Communications", IEEE Communications Magazine, Vol. 50, no. 12, pp. 36-43, Dec 2012

[21] Crabtree B., Stevens T., Allan B., Lederer S., Posch D., Mueller C., Timmerer C., Video Adaptation in Limited or Zero Network Coverage, CCNxConn 2013,PARC, Palo Alto, pp. 1-2, 2013

[22] Fiat A., Naor M., "Broadcast Encryption", in Advances in Cryptology (Crypto'93), volume 773 of Lecture Notes in Computer Science, pages 480-491. Springer Berlin / Heidelberg, 1994.

[23] Posch D., Hellwagner H., Schartner P., "On-Demand Video Streaming based on Dynamic Adaptive Encrypted Content Chunks", in Proceedings of the 8th International Workshop on Secure Network Protocols (NPSec'13), Los Alamitos, IEEE Computer Society Press, October, 2013.

[24] Montpetit M.J., Westphal C., Trossen D., "Network Coding Meets Information Centric Networks," in Proceedings of the workshop on Name-Oriented Mobility (NOM), jointly with ACM MobiHoc 2013, Hilton Head, SC, June 2013.

[25] Le Callet P., Moeller S. and Perkis A. (eds.), Qualinet White Paper on Definitions of Quality of Experience (2012). European Network on Quality of Experience in Multimedia

Systems and Services (COST Action IC 1003), Lausanne, Switzerland, Version 1.2, March 2013.

[26] Medard M., Kim M., ParandehGheibi M., Zeng W., Montpetit M.J., Quality of Experience for Multimedia Communications: Network Coding Strategies, Technical Report, MIT, 2012/3

[27] Montpetit M.J., Holtzman H., Chakrabarti K., Matijasevic M., Social video consumption: Synchronized viewing experiences across devices and networks, IEEE International Conference on Communications Workshops (ICC), 2013, 286-290

[28] Su K., Westphal C., On the Benefit of Information Centric Networks for Traffic Engineering, IEEE ICC Conference, June 2014

[29] Chanda A., Westphal C., Raychaudhuri D., Content Based Traffic Engineering in Software Defined Information Centric Networks, in IEEE INFOCOM Workshop NOMEN'13, April, 2013

[30] Castro M., Druschel P., Kermarrec A.-M., Nandi A., Rowstron A., Singh A. SplitStream: Highbandwidth multicast in cooperative environments. In Proceedings of ACM SOSP (2003).

[31] ATIS IPTV Interoperability Forum, ATIS IFF, http://www.atis.org/iif/deliv.asp

14. Authors' Addresses

Stefan Lederer, Christian Timmerer, Daniel Posch
Alpen-Adria University Klagenfurt
Universitaetsstrasse 65-67, 9020 Klagenfurt, Austria

Email: {firstname.lastname}@itec.aau.at

Cedric Westphal, Aytac Azgin. Shucheng (Will) Liu
Huawei
2330 Central Expressway, Santa Clara, CA95050, USA

Email: {cedric.westphal,aytac.azgin,liushucheng}@huawei.com

Christopher Mueller
bitmovin GmbH
Lakeside B01, 9020 Klagenfurt, Austria

Email: christopher.mueller@bitmovin.net

Andrea Detti
Electronic Engineering Dept.
University of Rome Tor Vergata
Via del Politecnico 1, Rome, Italy

Email: andrea.detti@uniroma2.it

Daniel Corujo,
Advanced Telecommunications and Networks Group
Instituto de Telecomunicacoes
Campus Universitario de Santiago
P-3810-193 Aveiro, Portugal

Email: dcorujo@av.it.pt

Jianping Wang
City University of Hong Kong
Hong Kong, China

Email: jianwang@cityu.edu.hk

Marie-Jose Montpetit

Email: marie@mjmontpetit.com

Niall Murray
Dept. of Electronic, Computer and Software Engineering
Athlone Institute of Technology
Dublin Rd., Athlone, Ireland

Email: nmurray@research.ait.ie

15. Acknowledgements

This work was supported in part by the EC in the context of the SocialSensor (FP7-ICT-287975) project and partly performed in the Lakeside Labs research cluster at AAU. SocialSensor receives research funding from the European Community’s Seventh Framework Programme. The work for this document was also partially performed

in the context of the FP7/NICT EU-JAPAN GreenICN project, http://www.greenicn.org. Apart from this, the European Commission has no responsibility for the content of this draft. The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability. The authors would like to thank Shucheng (Will) Liu from Huawei for supporting Dr. Wang’s research and Marie-Jose Montpetit of MIT for her help in writing the AAA for DRM section.

ICNRG
Internet-Draft
Intended status: Experimental
Expires: December 3, 2016

M. Mosko
PARC, Inc.

June 1, 2016

CCNx Content Object Chunking
draft-mosko-icnrg-ccnxchunking-02

Abstract

This document specifies a chunking protocol for dividing a user payload into CCNx Content Objects. This includes specification for the naming convention to use for the chunked payload and a field added to a Content Object to represent the last chunk of an object.

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 http://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 December 3, 2016.

Copyright Notice

Table of Contents

1. Introduction

CCNx Content Objects [CCNSemantics] are sized to amortize cryptographic operations over user data while simultaneously staying a reasonable size for transport over today's networks. This means a Content Object is usually within common UDP or jumbo Ethernet size. If a publisher has a larger amount of data to associate with a single Name, the data should be chunked with this chunking protocol. This protocol uses state in the Name and in an optional field within the Content Object. A chunked object may also have an external metadata content object that describes the original pre-chunked object.

CCNx uses two types of messages: Interests and Content Objects [CCNSemantics]. An Interest carries the hierarchically structured variable-length identifier (HSVLI), or Name, of a Content Object and serves as a request for that object. If a network element sees multiple Interests for the same name, it may aggregate those Interests. A network element along the path of the Interest with a matching Content Object may return that object, satisfying the Interest. The Content Object follows the reverse path of the Interest to the origin(s) of the Interest. A Content Object contains the Name, the object's Payload, and the cryptographic information used to bind the Name to the payload.

This specification adds a new segment to the Name TLV for conveying the chunk number. It updates [CCNMessages]. It also provides guidelines for the usage of the Key Locator in chunked objects.

Packets are represented as 32-bit wide words using ASCII art. Because of the TLV encoding and optional fields or sizes, there is no concise way to represent all possibilities. We use the convention that ASCII art fields enclosed by vertical bars "|" represent exact bit widths. Fields with a forward slash "/" are variable bitwidths, which we typically pad out to word alignment for picture readability.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Chunking

Chunking, as used in this specification, means serializing user data into one or more chunks, each encapsulated in a CCNx Content Object. A chunk is a contiguous byte range within the user data. One segment in the Name of that Content Object represents the chunk number. A field in the Content Object - only mandatory in the final chunk - represents the end of the stream. Chunks are denoted by a serial counter, beginning at 0 and incrementing by 1 for each contiguous chunk. The chunking ends at the final chunk. No valid user data exists beyond the final chunk, and reading beyond the final chunk MUST NOT return any user data.

Chunking MUST use a fixed block size, where only the final chunk MAY use a smaller block size. This is required to allow a reader to seek to a specific byte offset once it knows the block size. The blocksize may be inferred from the size of the first chunk of user data. The first chunk of user data may not be chunk 0.

Because of the requirement for a fixed block size, the inclusion of certain cryptographic fields in the same content objects as user data would throw off the ability to seek. Therefore, it is RECOMMENDED that all required cryptographic data, such as public keys or key name links, be included in the leading chunks before the first byte of user data. User data SHOULD then run continuously and with the same block size through the remainder of the content objects.

This draft introduces a new Name path segment TLV type, called the ChunkNumber name segment. The ChunkNumber name segment is the serial order of the chunks. It MUST begin at 0 and MUST be incremented by 1. The ChunkNumber name segment is appended to the base name of the user data, and is usually the last name segment.

The new Content Object field is the EndChunkNumber. It MUST be included in the Content Object which is the last chunk of user data, but SHOULD be present at the earliest time it is known. The value of the EndChunkNumber should be the network byte order value of the last ChunkNumber. For example, if 3000 bytes of user data is split with a 1200 byte block size, there will be 3 chunks: 0, 1, and 2. The EndChunkNumber is 2.

The EndChunkNumber may be updated in later Chunks to a larger value, as long as it has not yet reached the end. The EndChunkNumber SHOULD NOT decrease. If a publisher wishes to close a stream before reaching the End Chunk, it should publish empty Content Objects to fill out to the maximum EndChunkNumber ever published. These padding chunks MUST contain the true EndChunkNumber.

2.1. Cryptographic material

Chunk 0 SHOULD include the public key or key name link used to verify the chunked data. It is RECOMMENDED to use the same key for the whole set of chunked data. If a publisher uses multiple keys, then the public key or key name link for all keys SHOULD be in the leading chunks before any user data.

The rationel for putting all cryptographic data up front is because the protocol requires using a fixed block size for all user data to enable seeking in the chunked stream.

2.2. Examples

Here are some examples of chunked Names using the Labeled Content Identifier URI scheme in human readable form (ccnx:).

In this example, the content producer publishes a JPG that takes 4 Chunks. The EndChunkNumber is missing in the first content object (Chunk 0), but is known and included when Chunk 1 is published. It is omitted in Chunk 2, then appears in Chunk 3, where it is mandatory.

```
ccnx:/Name=parc/Name=picture.jpg/Chunk=0  --
ccnx:/Name=parc/Name=picture.jpg/Chunk=1  EndChunkNumber=3
ccnx:/Name=parc/Name=picture.jpg/Chunk=2  --
ccnx:/Name=parc/Name=picture.jpg/Chunk=3  EndChunkNumber=3
```

In this example, the publisher is writing an audio stream that ends before expected so the publisher fills empty Content Objects out to the maximum ChunkNumber, stating the correct EndChunkNumber. Chunks 4, 5, and 6 do not contain any new user data.

```
ccnx:/Name=parc/Name=talk.wav/Chunk=0  --
ccnx:/Name=parc/Name=talk.wav/Chunk=1  EndChunkNumber=6
ccnx:/Name=parc/Name=talk.wav/Chunk=2  --
ccnx:/Name=parc/Name=talk.wav/Chunk=3  EndChunkNumber=3
ccnx:/Name=parc/Name=talk.wav/Chunk=4  EndChunkNumber=3
ccnx:/Name=parc/Name=talk.wav/Chunk=5  EndChunkNumber=3
ccnx:/Name=parc/Name=talk.wav/Chunk=6  EndChunkNumber=3
```

3. TLV Types

This section specifies the TLV types used by CCNx chunking.

3.1. Name Types

CCNx chunking uses two new Name types: Chunk Number and Chunk Metadata.

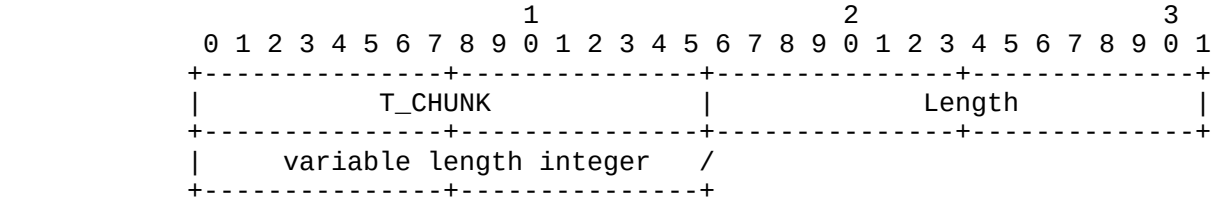
Type	Abbrev	Name	Description
%x0010	T_CHUNK	Chunk Number (Section 3.1.1)	The current Chunk Number, is an unsigned integer in network byte order without leading zeros. The value of zero is represented as the single byte %x00.

Table 1: Name Types

3.1.1. Chunk Number

The current chunk number, as an unsigned integer in network byte order without leading zeros. The value of zero is represented as the single byte %x00.

In ccnx: URI form, it is denoted as "Chunk".



3.2. Protocol Information

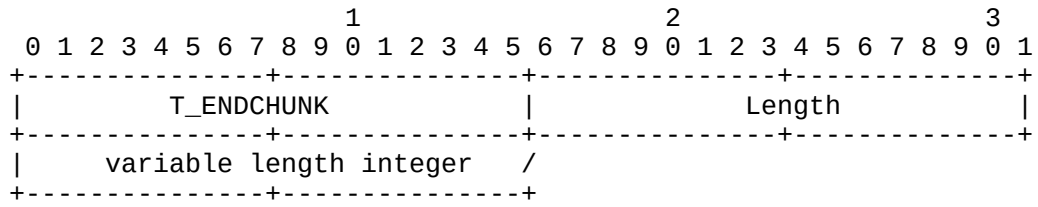
CCNx chunking introduces one new TLV for use in a Content Object.

Type	Abbrev	Name	Description
%x000C	T_ENDCHUNK	EndChunkNumber (Section 3.1.1)	The last Chunk number, as an unsigned integer in network byte order without leading zeros. The value of zero is represented as the single byte %x00.

Table 2: Content Object Types

3.2.1. EndChunkNumber

The ending chunk number, as an unsigned integer in network byte order without leading zeros. The value of zero is represented as the single byte %x00.



4. Acknowledgements

5. IANA Considerations

The draft adds new types to the CCNx Name Segment Types registry and the CCNx Content Object Types registry.

6. Security Considerations

This draft does not put any requirements on how chunked data is signed or validated.

7. References

7.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

7.2. Informative References

[CCNMessages] Mosko, M., Solis, I., and C. Wood, "CCNx Messages in TLV Format (Internet draft)", 2016, <http://tools.ietf.org/html/draft-irtf-icnrg-ccnxmessages-02>.

[CCNSemantics] Mosko, M., Solis, I., and C. Wood, "CCNx Semantics (Internet draft)", 2016, <http://tools.ietf.org/html/draft-mosko-icnrg-ccnxsemantics-03>.

[CCNx] PARC, Inc., "CCNx Open Source", 2007, <http://www.ccnx.org>.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, DOI 10.17487/RFC2616, June 1999, <http://www.rfc-editor.org/info/rfc2616>.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <http://www.rfc-editor.org/info/rfc3552>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <http://www.rfc-editor.org/info/rfc5226>.

Author’s Address

Marc Mosko
PARC, Inc.
Palo Alto, California 94304
USA

Phone: +01 650-812-4405
Email: marc.mosko@parc.com

ICNRG
Internet-Draft
Intended status: Experimental
Expires: October 8, 2016

M. Mosko
C. Wood
PARC, Inc.

April 6, 2016

The CCNx URI Scheme
draft-mosko-icnrg-ccnxurischeme-01

Abstract

This document defines an RFC3986 URI compliant identifier called a Labeled Segment URI in which name segments carry a label. This allows differentiation between unrelated resources with similar identifiers. This document also specifies the CCNx URI scheme, called "ccnx:," which conforms to the labeled segment encoding rules presented here. The CCNx URI scheme applies specific labels to each name segment of a URI to disambiguate between resources with similar names. This document defines a specific set of segment labels with label semantics.

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 http://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 October 8, 2016.

Copyright Notice

Table of Contents

1. Introduction

A Labeled Segment is an URI [RFC3986] compliant convention that allows an application or protocol to embed labels in name segments, thus disambiguating the resource identified by the path. Labeled Segment URIs also allow for query and fragment components to follow the Labeled Segment form.

Some protocols may wish to disambiguate name segments between different identifier spaces, such as "version" and "page". Other protocols may wish to use a type system such as "/str=parc/int=7" and "/str=parc/str=7". Labeled Segment URIs provide an unambiguous and flexible representation in systems that allow resources with otherwise similar names.

It is not sufficient to leave the determination of type to application-specific conventions. In a networked system with multiple applications accessing resources generated by other applications, there needs to be a set of common conventions. For example, if one application uses a base 64 encoding of a frame number, e.g. base64(0xbdea), and another uses "ver=" to represent a document version, there is an ambiguity because base64(0xbdea) is the string "ver=".

Labeled Segments defines "ls-segment" as "label[:param]=value", where the value only contains unreserved, percent-encoded, or certain sub-delim characters. In the previous example, one application would say "/frame=%BD%EA" and the other would say "/ver=".

In this document, we use URI [RFC3986] terminology, therefore a URI and CCNx Name are both composed of a URI path, which is a collection of name segments. We do not use the term "name component" as was common in old CCNx. In this document, the word "segment" alone means "name segment."

URIs conforming to the CCNx URI scheme carry a label for each name segment. The contents of each name segment must conform to the label semantics. Example segment types are "Binary Segment", "Name", and "KeyId".

We use Labeled Segment URIs as the canonical, human-readable representation. There is an unambiguous, one-to-one correspondence between an absolute LS-URI path and a Labeled Name. Relative URI representations are removed during encoding, so no relative name ends up in wire format. Some labels are URIs that are IRI [RFC3987] compatible.

Labeled Names shall be used everywhere a Name is used in CCNx, such

as in the Name of an Interest or Content Object. They are also used in Links, KeyLocators, or any other place requiring a name. When encoded for the wire, a binary representation is used, depending on the specific wire format codec, which is outside the scope of this document.

This document specifies:

- the ccnx scheme.
- a canonical URI representation.

Formal grammars use the ABNF [RFC5234] notation.

TODO: We have not adopted Requirements Language yet.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. URI path segment grammar for label=value pairs

2.1. Labeled Segments

This section describes the formal grammar for Labeled Segments using ABNF [RFC5234] notation. We do not impose restrictions on the length of labels or values. The semantics of values are URI scheme specific, here we only describe the meta-structure of Labeled Segments. We begin by reviewing some definitions from [RFC3986] that define an absolute path URI.

```
URI           = scheme ":" hier-part [ "?" query ] [ "#" fragment ]
hier-part     = "//" authority path-abempty
                / path-absolute
                / <other path types>
path-absolute = "/" [ segment-nz *( "/" segment ) ]
segment       = *pchar
segment-nz    = 1*pchar
pchar         = unreserved / pct-encoded / sub-delims / ":" / "@"
query         = *( pchar / "/" / "?" )
fragment      = *( pchar / "/" / "?" )
pct-encoded   = "%" HEXDIG HEXDIG
unreserved    = ALPHA / DIGIT / "-" / "." / "_" / "~"
reserved      = gen-delims / sub-delims
gen-delims    = ":" / "/" / "?" / "#" / "[" / "]" / "@"
sub-delims    = "!" / "$" / "&" / "'" / "(" / ")"
                    / "*" / "+" / "," / ";" / "="
```

Labeled Segments defines a new segment type that provides unambiguous representation of a segment's label and its value. We define the top-level LS-URI as the same form as a URI, wherein each part conforms to the Label Segment grammar, which is a subset of the URI grammar.

```
LS-URI           = scheme ":" ls-hier-part ["?" ls-query]
                   ["#" fragment]
ls-hier-part     = ["//" authority] ls-path-absolute
ls-path-absolute = "/" [ first-segment *( "/" ls-segment ) ]
first-segment    = ls-segment-nz
ls-segment       = lpv-segment / v-segment
lpv-segment      = label [":" param] "=" *s-value-nz
v-segment        = *s-value-nz
ls-segment-nz    = lpv-segment-nz / v-segment-nz
lpv-segment-nz   = label [":" param] "=" s-value-nz
v-segment-nz     = s-value-nz
label            = alpha-t / num-t
param            = alpha-t / num-t
s-value-nz       = 1*(s-pchar)

ls-query         = *1 ( lpv-component / v-component
                        *( "&" (lpv-component / v-component) ) )
lpv-component    = label [":" param] "=" q-value
v-component      = q-value
q-value          = *(q-pchar)

alpha-t          = ALPHA *(ALPHA / DIGIT)
num-t            = dec-t / hex-t
dec-t            = 1*(DIGIT)
hex-t            = "0x" 1*(HEXDIG)
ls-pchar         =  unreserved / pct-encoded / ls-sub-delims
s-pchar          = ls-pchar / ":" / "@" / "&"
q-pchar          = ls-pchar / ":" / "@" / "/"
ls-sub-delims    = "!" / "$" / "'" / "(" / ")"
                       / "*" / "+" / "," / ";"
```

A Labeled Segment URI (LS-URI) contains a scheme that uses Labeled Segments, an optional authority, a labeled segment absolute path (ls-path-aboslute), an optional labeled segment query (ls-query), and a fragment. The authority is URI scheme specific and the fragment is independent of the URI scheme.

the ls-path-aboslute is a first-segment followed by zero or more "/" ls-segment. The first-segment may be empty or a non-zero ls-segment (ls-segment-nz). If it is empty, it corresponds to a 0-lenght name which typically is a default route. It is distinct from a 1-segment name with no value (which is not allowed). The first-segment MUST either be empty (the 0-lenght name) or MUST have a value. If the first-segment is an ls-segment-nz, then it will have a value.

An ls-segment may (lpv-segment) or may not (v-segment) have a label. A particular LS-URI scheme MUST define how unlabeled segments are processed, and MAY disallow them. A v-segment is an implied type.

Once the implied type is resolved, it functions like an lpv-segment.

An lpv-segment has a label, optional parameter, and optional value (s-value-nz). An empty value is a 0-length name segment with a defined type. This is distinct from a 0-length first-segment, which has neither type nor value.

lpv-segment values come from the s-pchar set, which excludes the "=" equal sign. This means that the only equal sign in a name segment must be the delimiter between the label:param and the value. Within the value, an equal sign must be percent encoded.

lpv-segment labels and values may be alpha-numeric identifiers or numbers (decimal or hexadecimal). For example, one scheme may define the labels "name", "version", and "frame". A version may be of types "date" or "serial", meaning that the version is either a date or a monotonic serial number. Some examples of resulting LS-URIs are: "/name=parc/name=csl/version:date=20130930" or "/name=alice_smith/version:serial=299". The parameters may also indicate an instance of a label, such as "/name=books/year:1=1920/year:3=1940", where there are scheme or application semantics associated with "year:1" and "year:3".

lpv-segment labels and parameters may also be numbers. For example, a protocol with a binary and URI representation may not have pre-defined all possible labels. In such cases, it could render unknown labels as their binary value, such as "/name=marc/x2003=green".

The ls-query component is a non-hierarchical set of components separated by "&". Each ls-query component is either a lpv-component or a v-component, similar to segments. They are based on q-value, which uses q-pchar that excludes "&", but includes "/". This allows an LS-URI scheme to use type query parameters.

Labeled Segments allow for dot-segments "." and ".." in a v-segment. They operate as normal. A single dot "." refers to the current hierarchy level and may be elided when the URI is resolved. Double dot ".." segments pop off the previous non-dot segment. An lpv-segment with a value of "." or ".." is not a dot-segment. It means that the value of the given label is "." or "..". For example /a=parc/b=csl/.. is equivalent to "/a=parc/b=csl", but the LS-URI "/a=parc/b=csl/c=.." does not contain a dot-segment.

2.2. URI comparison

An LS-URI scheme MUST specify the normalization rules to be used, following the methods of Section 6 [RFC3986]. At minimum, an LS-URI scheme SHOULD do the following:

- Normalize unrestricted percent-encodings to the unrestricted form.
- Normalize num-t to either dec-t or hex-t.
- If the scheme allows for value-only segments or query components and interprets them as a default type, they should be normalized to having the type specified.
- If the scheme allows for undefined labels and represents them, for example, as num-t, then it should normalize all labels to their corresponding num-t. If "name", for example, is known to be %x50 in a binary encoding of the URI, then all labels should be compared using their numeric value.

3. Application to CCNx Names

3.1. The ccnx Scheme

This section describes the CCNx URI scheme "ccnx:" for Labeled Names. A Labeled Name assigns a semantic type or label to each segment of the hierarchical content Name.

Unless otherwise specified, a name segment is an arbitrary sequence of octets.

Several name segment labels are binary unsigned integers. These are always encoded as variable length sequences of 1 or more octets in network byte order using the shortest representation (i.e. no leading %x00). The value of "0" is encoded as the single byte of "%x00". A zero-length sequence must be interpreted as "not present."

The CCNx Name segment types are:

- Name Segment: A generic name segment that includes arbitrary octets.

- Application Type N: An application may use application-specific parameters, numbered as integers, where N is from 0 to a system-specific maximum, not less than 255. These are represented as "App:1=value", for example.

It is common for an information centric networking protocol, such as CCNx or NDN, to use a binary on-the-wire representation for messages. Such protocols, if they use the ccnx: scheme, must have an appropriate codec that unambiguously represents Labeled Content Information in the chosen wire format. Relative dot-segments should not occur in the wire format, they should be resolved before encoding.

3.2. URI Representation

Typed Names use a standard RFC 3986 representation following the LS-URI convention. A name segment consists of any "unreserved" characters plus percent-encoded characters. Reserved characters must be percent encoded.

Within an absolute path, each segment consists of an "ls-segment" (c.f. LS-URI). A labeled segment is a type and a name component value, with a URI representation of "type=value". The "type=" portion may be omitted if it is type Name.

Some name types take a parameter, such as the Application types.

They are represented as "A:nnn=value", where the "nnn" is the application type number and value is the name component.

A CCNx URI MUST NOT include an Authority, Query, or Fragment. It is an error to include them.

Dot-segments (relative name components) are resolved when the URI is converted to a Typed Name. The "." dot-segment is removed. The ".." dot-segment is removed along with the previous non-dot-segment.

Type	Display	Name
'Name'	Hexadecimal	Name Segment
'IPID'	Hexadecimal	Interest Payload Identifier segment
'App:0' - 'App:255'	Hexadecimal	Application Component

Table 1: The CCNx URI Scheme Types

3.2.1. Examples

A name / is
ccnx:/ and is a 0-length name.

A name /Name= is
ccnx:/Name= and is a 1-segment name of 0-length.

A name /foo/bar.
ccnx:/Name=foo/Name=bar
ccnx:/foo/Name=bar
ccnx:/foo/bar

A name /foo/bar with key %xA0.
ccnx:/Name=foo/Name=bar/App:1=0xA0

A name /foo/bar with version %xA0 and App:2 value 0x09.
ccnx:/foo/bar/Version=0xA0/App:2=0x09

A name /foo/.., where the ".." is a literal name component,
not a relative dot-segment.
ccnx:/foo/Name=..

A name /foo/bar with application type 0 "hello"
and application type 1 "world".
ccnx:/Name=foo/Name=bar/App:0=hello/App:1=world

3.3. ccnx: URI comparison

While most comparisons are done using a wire format representation of a ccnx: URI, some applications may compare the CCNx URI using their URI representation. This section defines the rules for comparing ccnx: URIs using the methods of Section 6 [RFC3986]

Comparing typed name URIs must be done with:

- Syntax-based normalization
- Case normalization: normalize the representation of percent encodings. ccnx: does not use the host portion of the URI, and should be ignored if present.
- Percent encoding normalization: Percent encodings of unreserved characters must be converted to the unreserved character.
- Path segment normalization: dot-segments must be resolved first.

- Scheme-based normalization: The authority should be removed and the path represented as an absolute path.

- Protocol-based normalization: Should not be done. A trailing slash indicates a zero-length terminal name component and signifies a different name.

- typed-name-segment normalization: All segments should be presented with their type, do not elide the "N=" for Name components.

- Binary unsigned integer normalization: remove any leading %x00 from numbers, leaving only the terminal %x00 for "0".

- type parameters: they must have their percent encodings normalized. If they are integers, such as for the ’A’ type, they must not have leading zeros.

4. IRI Considerations

International Resource Identifiers extend the unreserved character set to include characters above U+07F and encode them using percent encoding. This extension is compatible with the ccnx: schema. It applies only to the "value" portion of an ls-segment.

The canonical name is determined by the URI representation of the IRI, after applying the rules of Section 3.1 of [RFC3987] and resolving dot-segments. The canonical name thus includes the URI representation of language markers, including the bidirectional components.

The value of a UTF-8 Name segment should be interpreted using IRI rules, including bidirectional markers. They may be displayed using localized formats.

Binary unsigned integer types are not interpreted under IRI rules, they are specifically percent encoded numbers. They may be displayed using a localized format.

5. Acknowledgements

6. IANA Considerations

This memo includes no request to IANA.

All drafts are required to have an IANA considerations section (see Guidelines for Writing an IANA Considerations Section in RFCs [RFC5226] for a guide). If the draft does not require IANA to do anything, the section contains an explicit statement that this is the case (as above). If there are no requirements for IANA, the section will be removed during conversion into an RFC by the RFC Editor.

7. Security Considerations

All drafts are required to have a security considerations section. See RFC 3552 [RFC3552] for a guide.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

8.2. Informative References

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <http://www.rfc-editor.org/info/rfc3552>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <http://www.rfc-editor.org/info/rfc3986>.

[RFC3987] Duerst, M. and M. Suignard, "Internationalized Resource Identifiers (IRIs)", RFC 3987, DOI 10.17487/RFC3987, January 2005, <http://www.rfc-editor.org/info/rfc3987>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <http://www.rfc-editor.org/info/rfc5226>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <http://www.rfc-editor.org/info/rfc5234>.

Authors' Addresses

Marc Mosko
PARC, Inc.
Palo Alto, California 94304
USA

Phone: +01 650-812-4405
Email: marc.mosko@parc.com

Christopher A. Wood
PARC, Inc.
Palo Alto, California 94304
USA

Phone: +01 650-812-4421
Email: christopher.wood@parc.com

ICNRG Working Group
Internet-Draft
Intended status: Experimental
Expires: September 14, 2017

C. Tschudin
University of Basel
C. Wood
University of California Irvine

March 13, 2017

File-Like ICN Collection (FLIC)
draft-tschudin-icnrg-flic-03

Abstract

This document describes a bare bones "index table"-approach for organizing a set of ICN data objects into a large, File-Like ICN Collection (FLIC).

At the core of this collection is a so called manifest which acts as the collection's root node. The manifest contains an index table with pointers, each pointer being a hash value pointing to either a final data block or another index table node.

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 http://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 14, 2017.

Table of Contents

1. Introduction

1.1. FLIC as a Distributed Data Structure

One figure

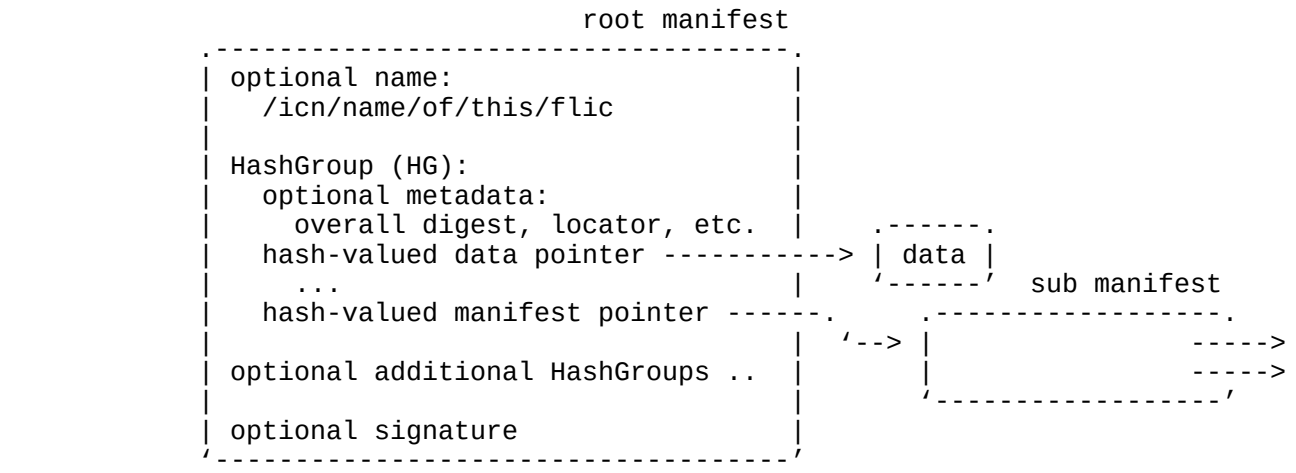


Figure 1: A FLIC manifest and its directed acyclic graph

1.2. Design goals

- Copy the proven UNIX inode concept:
  - index tables and memory pointers
- Adaption to ICN:
  - hash values instead of block numbers, unique with high probability
- Advantages (over non-manifest collections):
  - single root manifest signature covers all elements of the full collection, including intermediate sub manifests
  - eliminate reference to chunk numbering schemata (hash values only)
  - supports block-level de-duplication (can lead to a directed acyclic graph, or DAG, instead of a tree)
- Limitations
  - All data leafs must be present at manifest creation time (otherwise one cannot compute the pointers)
- Potential extensions (for study):

* Enhance the manifest such that it can serve as a "database cursor" or as a cursor over a time series, e.g. having entries for "previous" and "next" collections.

2. File-Like ICN Collection (FLIC) Format

We first give the FLIC format in EBN notation:

```
ManifestMsg := Name? HashGroup+

HashGroup   := MetaData? (SizeDataPtr | SizeManifestPtr)+
BlockHashGroup := MetaData? SizePerPtr (DataPtr | ManifestPtr)+

DataPtr := HashValue
ManifestPtr := HashValue
SizeDataPtr := Size HashValue
SizeManifestPtr := Size HashValue

SizePerPtr    := Size
HashValue     := See {{CCNxMessages}}
Size          := OCTET[8]

MetaData    := Property*
Property    := Locator | OverallByteCount | OverallDataDigest | ...
```

Description:

- The core of a manifest is the sequence of "hash groups".
- A HashGroup (HG) consists of a sequence of "sized" data or manifest pointers.
- A BlockHashGroup (BHG) consists of a sequence of data or manifest pointers and a mandatory field that lists the total size of each pointer. These HashGroups should be used when each pointer (except the last) contains an identical number of application bytes.
- Sizes are 64-bit unsigned integers.
- Data and manifest pointers are cryptographic HashValues encoded according to the mechanism listed in [CCNxMessages]. Specifically, a HashValue specifies the cryptographic hash algorithm and the actual digest.
- A HashGroup can contain a metadata section to help a reader to optimize content retrieval (block size of leaf nodes, total size, overall digest etc).

- None of the ICN objects used in FLIC are allowed to be chunked, including the (sub-) manifests. The smallest possible complete manifest contains one HashGroup with one pointer to an ICN object.

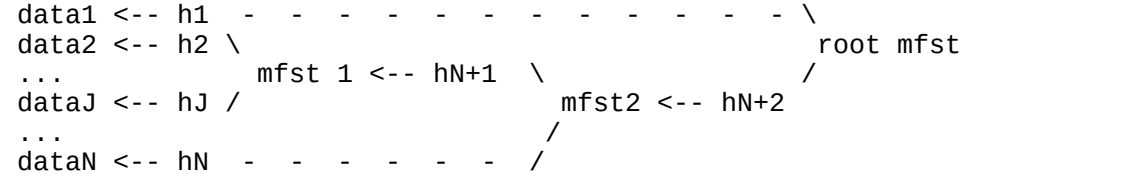
2.1. Use of hash-valued pointers

FLIC's tree data structure is a generalized index table as it is known from file systems. The pointers, which in an OS typically are hard disk block numbers, are replaced by hash values of other ICN objects. These ICN objects contain either other manifest nodes, or leaf nodes. Leafs contain the actual data of the collection. Each pointer explicitly indicates the amount of application data bytes contained by the referred object. For example, the size of a data pointer (to a leaf) represents the size of the leaf's content object payload. Conversely, the size of a manifest pointer represents the total size of all pointers contained in that manifest.

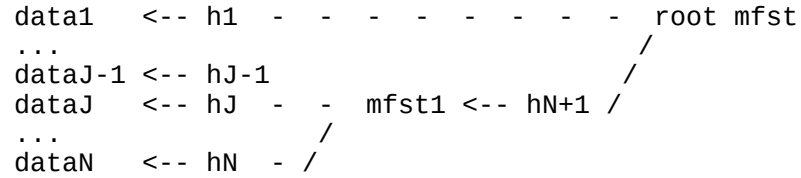
FLIC makes use of "nameless ICN object" where the network is tasked with fetching an object based on its digest only. The interest for such an object consists of a routing hint (locator) plus the given digest value.

2.2. Creating a FLIC data structure

Starting from the original content, the corresponding byte array is sliced into chunks. Each chunk is encoded as a data object, according the ICN suite. For each resulting data object, the hash value is computed. Groups of consecutive objects are formed and the corresponding hash values collected in manifests, which are also encoded. The hash values of the manifest objects replace the hash values of the covered leaf nodes, thus reducing the number of hash values. This process of hash value collection and replacement is repeated until only one (root) manifest is left.



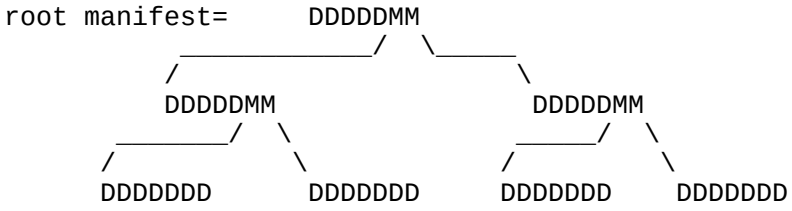
Of special interest are "skewed trees" where a pointer to a manifest may only appear as last pointer of (sub-) manifests. Such a tree becomes a sequential list of manifests with a maximum of datapointers per manifest packet. Beside the tree shape we also show this data structure in form of packet content where D stands for a data pointer and M is the hash of a manifest packet.



A pseudo code description for producing a skewed tree follows below.

```
Input:
    Application data D of size |D| (bytes)
    Block size B (in bytes)
Output:
    FLIC root node R
Algo:
    n = number of leaf nodes = ceil(|D| / B)
    k = number of (encoded) hash values fitting in a block of size B
    H[1..n] = array of hash values
      initialized with the data hash values for data chunks 1..n
    While n > k do
      a)  create manifest M with a HashGroup
      b)  append to the HashGroup in M all hash values H[n-k+1..n]
      c)  n = n - k + 1
      d)  H[n] = manifest hash value of M
    Create root manifest R with a HashGroup
    Add to the HashGroup of R all hash values H[1..n]
    Optionally: add name to R, sign manifest R
    Output R
```

Obtaining with each manifest a maximum of data pointers is beneficial for keeping the download pipeline filled. On the other hand, this tree doesn't support well random access to arbitrary byte positions: All data pointers coming before that offset have to be fetched before locating the block of interest. For random access, binary trees (where both subtrees of a node cover half of the content bytes) are better suited. This can be combined with the "skewed tree" approach: Manifests of intermediate nodes are filled with data pointers except for the last two slots. The second last slot points to a manifest for the "first half" of the left content, the last slots then points to a manifest for the rest.



This can be generalized to k-ary trees by allocating k pointers per manifest instead of 2.

2.3. Reconstructing the collection’s data

To fetch the data associated with a given FLIC (sub-) manifest, the receiver sequentially works through all entries found in the HashGroups and issues corresponding hash-based interests. In case of a data hash pointer, the received content object is appended. In case of a manifest hash pointer, this procedure is called recursively for the received manifest. In other words, the collection data is represented as the concatenation of data leaves from this _pre-order_ depth-first search (DFS) traversal strategy of the manifest tree. (Currently, pre-order DFS is the only supported traversal strategy.) This procedure works regardless of the tree’s shape.

A pseudo code description for fetching is below.

```
Input:
    Root manifest R
Output:
    Application data D
Algo:
    global D = []
    DFS(R)
    Output D

where:

procedure DFS(M)
{
L:
  H = sequence of hash valued pointers of M
  foreach p in H do:
    if p is a data pointer then
      data = lookup(p)
      Append data to D
    else
      M = lookup(p)
      if p is last element in H then
        goto L;   // tail recursion
      DFS(M)
}
```

The above DFS code works for FLIC manifest trees of arbitrary shape. In case of a skewed tree, no recursion is needed and a single instance of the DFS procedure suffices (i.e., one uses tail recursion).

2.4. Metadata in HashGroups

In FLIC, metadata is linked to HashGroups and permits to inform the FLIC retriever about properties of the data that is covered by this hash group. Examples are overall data bytes or the overall hash digest (this is akin to a Merkle hash). The intent of such metadata is to enable an in-network retriever to optimize its operation - other attributes linked to the collection as a whole (author, copyright, etc.) is out of scope.

The list of available metadata is below.

* Locator - provides a new routing hint (name prefix) where the chunks of this hash group can be retrieved from. The default is to use the locator of the root manifest.

* OverallByteCount - indicates the total number of *application data bytes* contained in a single HashGroup. This does not include bytes consumed by child manifests. This value is equal to the sum of all pointer sizes contained in the HashGroup.

* OverallDataDigest - expresses the overall digest of all application data contained in the HashGroup.

BlockHashGroups contain a mandatory piece of metadata called the SizePerPtr. This value indicates the total number of application bytes contained within each pointer in the hash group _except for the last pointer._ Normal HashGroups do not require this piece of metadata; Instead, each pointer includes their size explicitly.

2.5. Locating FLIC leaf and manifest nodes

The optional name of a manifest is a mere decoration and has no locator functionality at all: All objects pointed to by a manifest are retrieved from the location where the manifest itself was obtained from (which is not necessarily its name). Example:

```
Objects:
  manifest(name=/a/b/c, ptr=h1, ptr=hN)  - has hash h0
  nameless(data1)                        - has hash h1
  ...
  nameless(dataN)                        - has hash hN

Query for the manifest:
  interest(name=/the/locator/hint, implicitDigest=h0)
```

In this example, the name "/a/b/c" does NOT override "/the/locator/hint" i.e., after having obtained the manifest, the retriever will issue requests for

```
  interest(name=/the/locator/hint, implicitDigest=h1)
  ...
  interest(name=/the/locator/hint, implicitDigest=hN)
```

Using the locator metadata entry, this behavior can be changed:

```
Objects:
  manifest(name=/a/b/c,
           hashgroup(loc=/x/y/z, ptr=h1)
           hashgroup(ptr=h2)             - has hash h0
  nameless(data1)                        - has hash h1
  nameless(data2)                        - has hash h2

Queries:
  interest(name=/the/locator/hint, implicitDigest=h0)
  interest(name=/x/y/z, implicitDigest=h1)
  interest(name=/the/locator/hint, implicitDigest=h2)
```

3. Advanced uses of FLIC manifests

The FLIC mechanics has uses cases beyond keeping together a set of data objects, such as: seeking, block-level de-duplication, re-publishing under a new name, growing ICN collections, and supporting FLICs with different block sizes.

3.1. Seeking

Fast seeking (without having to sequentially fetch all content) works by skipping over entries for which we know their size. The following expression shows how to compute the byte offset of the data pointed at by pointer P_i, offset_i. In this formula, let P_i represent the Size value of the i-th pointer.

$$offset_i = \sum_{i = 1}^{i - 1} P_i.size$$

With this offset, seeking is done as follows:

```
Input: seek_pos P, a FLIC manifest with a hash group having N entries
Output: pointer index i and byte offset o, or out-of-range error
Algo:
    offset = 0
    for i in 1..N do
        if (P < P_i.size)
            return (i, P - offset)
        offset += P_i.size
    return out-of-range
```

Seeking in a BlockHashGroup is different since offsets can be quickly computed. This is because the size of each pointer P_i except the last is equal to the SizePerPtr value. For a BlockHashGroup with N pointers, OverallByteCount D, and SizePerPointer L, the size of P_i is equal to the following:

$$D - ((i - 1) * L)$$

In a BlockHashGroup with k pointers, the size of P_k is equal to:

```
D - L * (k - 1)
```

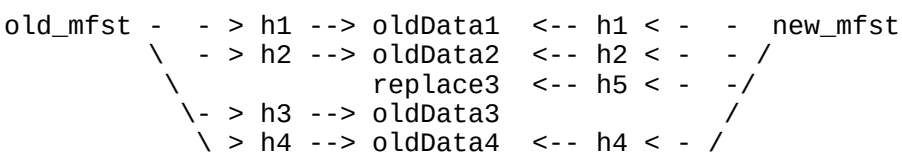
Using these, the seeking algorithm can be thus simplified to the following:

```
Input: seek_pos P, a FLIC manifest with a hash group having
       OverallByteCount S and SizePerPointer L.
Output: pointer index i and byte offset o, or out-of-range error
Algo:
    if (P > S)
        return out-of-range
    i = floor(P / L)
    if (i > N)
        return out-of-range # bad FLIC encoding
    o = P mod L
    return (i, o)
```

Note: In both cases, if the pointer at position i is a manifest pointer, this algorithm has to be called once more, seeking to seek_pos o inside that manifest.

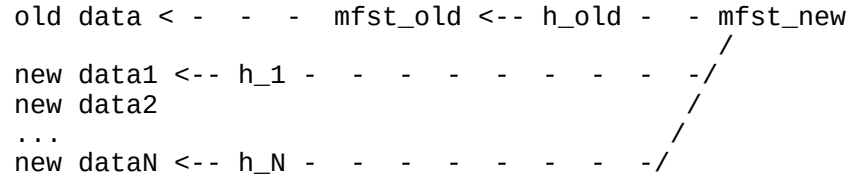
3.2. Block-level de-duplication

Consider a huge file, e.g. an ISO image of a DVD or program in binary form, that had previously been FLIC-ed but now needs to be patched. In this case, all existing encoded ICN chunks can remain in the repository while only the chunks for the patch itself is added to a new manifest data structure, as is shown in the picture below. For example, the venti [1] archival file system of Plan9 uses this technique.



3.3. Growing ICN collections

A log file, for example, grows over time. Instead of having to re-FLIC the grown file it suffices to construct a new manifest with a manifest pointer to the old root manifest plus the sequence of data hash pointers for the new data (or additional sub-manifests if necessary). Note that this tree will not be skewed (anymore).



3.4. Re-publishing a FLIC under a new name

It can happen that a publisher's namespace is part of a service provider's prefix. When switching provider, the publisher may want to republish the old data under a new name. This can easily be achieved with a single nameless root manifest for the large FLIC plus arbitrarily many per-name manifests (which are signed by whomever wants to publish this data):

```
data < - nameless_mfst() <-- h  < - mfst(/com/parc/east/the/flic)
                                < - mfst(/com/parc/west/old/the/flic)
                                < - mfst(/internet/archive/flic234)
```

Note that the hash computation (of h) only requires reading the nameless root manifest, not the entire FLIC.

This example points out the problem of HashGroups having locator metadata elements: A retriever would be urged to follow these hints which are "hardcoded" deep inside the FLIC but might have become outdated. We therefore recommend to name FLIC manifests only at the highest level (where these names have no locator function). Child nodes in a FLIC manifest should not be named as these names serve no purpose except retrieving a sub-tree's manifest by name, if would be required.

3.5. Data Chunks of variable size

If chunks do not have regular (block) sizes, the HashGroup can be used to still convey to a reader the length of the chunks at the manifest level. (This can be computed based on the size of pointers, but the metadata field makes this determination simpler.) Example use cases would be chunks each carrying a single ASCII line as entered by a user or a database with variable length records mapped to chunks.

```
M = (manifest
      (hashgroup((metadata(SizePerPtr=4096)) (dataptr=h1))
      (hashgroup((metadata(SizePerPtr=1500)) (dataptr=h2))
      ...
    )
```

4. Encoding

We express the packet encoding of manifests in a symbolic expression style in order to show the TLV structure and the chosen type values. In this notation, a TLV's type is a combination of "SymbolicName/ Tvalue", Length is not shown and Values are sub-expressions. Moreover, we populate the data structure with all possible entries and omit repetition.

4.1. Example Encoding for CCNx1.0

```
[FIXED_HEADER OCTET[8]]
(ManifestMsg/T_MANIFEST
  (Name/T_NAME ...)
  (HashGroup/T_HASHGROUP
     (MetaData/T_HASHGROUP_METADATA
        (HGLocator/T_HASHGROUP_METADATA_LOCATOR (T_NAME ...))
        (HGOverallByteCount/T_HASHGROUP_METADATA_BYTECOUNT INT)
        (HGOverallDataDigest/T_HASHGROUP_METADATA_DATADIGEST OCTET[32])
     )
     (SizeDataPtr/T_HASHGROUP_SIZEDATAPTR OCTET[8] (T_HASH ...))
     (SizeMfstPtr/T_HASHGROUP_SIZEMANIFESTPTR OCTET[8] (T_HASH ...))
  )
  (BlockHashGroup/T_BLOCKHASHGROUP
     (MetaData/T_HASHGROUP_METADATA (...))
     (DataPtr/T_HASHGROUP_DATAPTR OCTET[32] (T_HASH ...))
     (MfstPtr/T_HASHGROUP_MANIFESTPTR OCTET[32] (T_HASH ...))
  )
)
```

Interest: name is locator, use objHashRestriction as selector.

4.2. Example Encoding for NDN

The assigned NDN content type value for FLIC manifests is 1024 (0x400).

```
(Data/0x6
  (Name/0x7 ...)
  (MetaInfo/0x14
    (ContentType/0x18 0x0400)
  )
  (Content/0x15
    (HashGroup/0xC0
      (MetaInfo/0x14
        (LocatorNm/0xC3 (NameComp/0x8 ...))
        (OverallDataDigest/0xC4 OCTET[32])
        (OverallByteCount/0xC5 INT)
      )
      (DataPtr/0xC1 OCTET[8] OCTET[32])
      (MfstPtr/0xC2 OCTET[8] OCTET[32])
      (SizeDataPtr/0xC3 OCTET[32])
      (SizeMfstPtr/0xC4 OCTET[32])
    )
  )
  (SignatureInfo/0x16 ...)
  (SignatureValue/0x17 ...)
)
```

Interest: name is locator, use implicitDigest name component as selector.

5. Security Considerations

The payloads of FLIC leaf nodes may be encrypted prior to construction. This does not have any impact on the FLIC construction process since all data is treated as if it were opaque.

6. References

6.1. Normative References

[CCNxMessages]
PARC, Inc, ., LinkedIn, ., and C. Wood, "CCNx Messages in TLV Format", n.d., <https://datatracker.ietf.org/doc/draft-irtf-icnrg-ccnxmessages/>.

6.2. URIs

[1] http://plan9.bell-labs.com/sys/doc/venti/venti.pdf

Authors' Addresses

Christian Tschudin
University of Basel

Email: christian.tschudin@unibas.ch

Christopher A. Wood
University of California Irvine

Email: woodc1@uci.edu

ICN Research Group
Internet-Draft
Intended status: Informational
Expires: October 24, 2016

Y. Zhang
D. Raychadhuri
WINLAB, Rutgers University
L. Grieco
Politecnico di Bari (DEI)
E. Baccelli
INRIA
J. Burke
UCLA REMAP
R. Ravindran
G. Wang
Huawei Technologies
A. Lindren
B. Ahlgren
SICS Swedish ICT
O. Schelen
Lulea University of Technology

April 22, 2016

Requirements and Challenges for IoT over ICN
draft-zhang-icnrg-icniot-requirements-01

Abstract

The Internet of Things (IoT) promises to connect billions of objects to the Internet. After deploying many stand-alone IoT systems in different domains, the current trend is to develop a common, "thin waist" of protocols forming a horizontal unified, defragmented IoT platform. Such a platform will make objects accessible to applications across organizations and domains. Towards this goal, quite a few proposals have been made to build a unified host-centric IoT platform as an overlay on top of today’s host-centric Internet. However, there is a fundamental mismatch between the host-centric nature of todays Internet and the information-centric nature of the IoT system. To address this mismatch, we propose to build a common set of protocols and services, which form an IoT platform, based on the Information Centric Network (ICN) architecture, which we call ICN-IoT. ICN-IoT leverages the salient features of ICN, and thus provides seamless mobility support, security, scalability, and efficient content and service delivery.

This draft describes representative IoT requirements and ICN challenges to realize a unified ICN-IoT framework. Towards this, we first identify a list of important requirements which a unified IoT architecture should have to support tens of billions of objects, then we discuss how the current IP-IoT overlay fails to meet these requirements, followed by discussion on suitability of ICN for IoT.

Though we see most of the IoT requirements can be met by ICN, we discuss specific challenges ICN has to address to satisfy them. Then we provide discussion of popular IoT scenarios including the "smart" home, campus, grid, transportation infrastructure, healthcare, Education, and Entertainment for completeness, as specific scenarios requires appropriate design choices and architectural considerations towards developing an ICN-IoT solution.

Status of This Memo

Copyright Notice

Table of Contents

1. IoT Motivation

During the past decade, many standalone Internet of Things (IoT) systems have been developed and deployed in different domains. The recent trend, however, is to evolve towards a globally unified IoT platform, in which billions of objects connect to the Internet, available for interactions among themselves, as well as interactions with many different applications across boundaries of administration and domains. Building a unified IoT platform, however, poses great challenges on the underlying network and systems. To name a few, it needs to support 50-100 Billion networked objects [1], many of which are mobile. The objects will have extremely heterogeneous means of

connecting to the Internet, often with severe resource constraints. Interactions between the applications and objects are often real-time and dynamic, requiring strong security and privacy protections. In addition, IoT applications are inherently information centric (e.g., data consumers usually need data sensed from the environment without any reference to the sub-set of motes that will provide the asked information). Taking a general IoT perspective, we first discuss the IoT requirements generally applicable to many well known scenarios. We then discuss how the current IP overlay models fail to meet these requirements. We follow this by key ICN features that makes it a better candidate to realize an unified IoT framework. We then discuss IoT challenges from an ICN perspective and requirements posed towards its design. Final discussion focuses on IoT scenarios and their unique challenges.

2. IoT Architectural Requirements

A unified IoT platform has to support interactions among a large number of mobile devices across the boundaries of organizations and domains. As a result, it naturally poses stringent requirements in every aspect of the system design. Below, we outline a few important requirements that a unified IoT platform has to address.

2.1. Naming

The first step towards realizing a unified IoT platform is the ability to assign names that are unique within the scope and lifetime of each device, data items generated by these devices, or a group of devices towards a common objective. Naming has the following requirements: first, names need to be persistent (within one or more contexts) against dynamic features that are common in IoT systems, such as lifetime, mobility or migration; second, names need to be secure based on application requirements; third, names should provide advantages to application authors in comparison with traditional host address based schemes.

2.2. Scalability

Cisco predicts there will be around 50 Billion IoT devices such as sensors, RFID tags, and actuators, on the Internet by 2020 [1]. As mentioned above, a unified IoT platform needs to name every entity such as data, device, service etc. Scalability has to be addressed at multiple levels of the IoT architecture including naming, security, name resolution, routing and forwarding level. In addition, mobility adds further challenge in terms of scalability. Particularly with respect to name resolution the system should be able to register/update/resolve a name within a short latency.

2.3. Resource Constraints

IoT devices can be broadly classified into two groups: resource-sufficient and resource-constrained. In general, there are the following types of resources: power, computing, storage, bandwidth, and user interface.

Power constraints of IoT devices limit how much data these devices can communicate, as it has been shown that communications consume more power than other activities for embedded devices. Flexible techniques to collect the relevant information are required, and uploading every single produced data to a central server is undesirable. Computing constraints limit the type and amount of processing these devices can perform. As a result, more complex processing needs to be conducted in cloud servers or at opportunistic points, example at the network edge, hence it is important to balance local computation versus communication cost.

Storage constraints of the IoT devices limit the amount of data that can be stored on the devices. This constraint means that unused sensor data may need to be discarded or stored in aggregated compact form time to time. Bandwidth constraints of the IoT devices limit the amount of communication. Such devices will have the same implication on the system architecture as with the power constraints; namely, we cannot afford to collect single sensor data generated by the device and/or use complex signaling protocols.

User interface constraints refer to whether the device is itself capable of directly interacting with a user should the need arise (e.g., via a display and keypad or LED indicators) or requires the network connectivity, either global or local, to interact with humans.

The above discussed device constraints also affect application performance with respect to latency and jitter. This in particular applies to satellite or other space based devices.

2.4. Traffic Characteristics

IoT traffic can be broadly classified into local area traffic and wide area traffic. Local area traffic is between nearby devices. For example, neighboring cars may work together to detect potential hazards on the highway, sensors deployed in the same room may collaborate to determine how to adjust the heating level in the room. These local area communications often involve data aggregation and filtering, have real time constraints, and require fast device/data/service discovery and association. At the same time, the IoT platform has to also support wide area communications. For example,

in Intelligent Transportation Systems, re-routing operations may require a broad knowledge of the status of the system, traffic load, availability of freights, whether forecasts and so on. Wide area communications require efficient data/service discovery and resolution services.

While traffic characteristics for different IoT systems are expected to be different, certain IoT systems have been analyzed and shown to have comparable uplink and downlink traffic volume in some applications such as [2], which means that we have to optimize the bandwidth/energy consumption in both directions. Further, IoT traffic demonstrates certain periodicity and burstiness [2]. As a result, when provisioning the system, the shape of the traffic volume has to be properly accounted for.

2.5. Contextual Communication

Many IoT applications shall rely on dynamic contexts in the IoT system to initiate communication between IoT devices. Here, we refer to a context as attributes applicable to a group of devices that share some common features, such as their owners may have a certain social relationship or belong to the same administrative group, or the devices may be present in the same location. For example, cars traveling on the highway may form a "cluster" based upon their temporal physical proximity as well as the detection of the same event. These temporary groups are referred to as contexts. IoT applications need to support interactions among the members of a context, as well as interactions across contexts.

Temporal context can be broadly categorized into two classes, long-term contexts such as those that are based upon social contacts as well as stationary physical locations (e.g., sensors in a car/ building), and short-term contexts such as those that are based upon temporary proximity (e.g., all taxicabs within half a mile of the Time Square at noon on Oct 1, 2013). Between these two classes, short-term contexts are more challenging to support, requiring fast formation, update, lookup and association.

2.6. Handling Mobility

There are several degrees of mobility in a unified IoT platform, ranging from static as in fixed assets to highly dynamic in vehicle-to-vehicle environments.

Mobility in the IoT platform can mean 1) the data producer mobility (i.e., location change), 2) the data consumer mobility, 3) IoT Network mobility (e.g., a body-area network in motion as a person is walking); and 4) disconnection between the data source and

destination pair (e.g., due to unreliable wireless links). The requirement on mobility support is to be able to deliver IoT data below an application's acceptable delay constraint in all of the above cases, and and if necessary to negotiate different connectivity or security constraints specific to each mobile context.

2.7. Storage and Caching

Storage and caching plays a very significant role depending on the type of IoT ecosystem, also a function subjected to privacy and security guidelines. In a unified IoT platform, depending on application requirements, content caching may or may not be policy driven. If caching is pervasive, intermediate nodes don't need to always forward a content request to its original creator; rather, locating and receiving a cached copy is sufficient for IoT applications. This optimization can greatly reduce the content access latencies.

Furthermore considering hierarchical nature of IoT systems, ICN architectures enable a more flexible, heterogeneous and potentially fault-tolerant approach to storage providing persistence at multiple levels.

In network storage and caching, however, has the following requirements on the IoT platform. The platform needs to support the efficient resolution of cached copies. Further the platform should strive for the balance between caching, content security/privacy, and regulations.

2.8. Security and Privacy

In addition to the fundamental challenge of trust management, a variety of security and privacy concerns also exist in ICNs.

The unified IoT platform makes physical objects accessible to applications across organizations and domains. Further, it often integrates with critical infrastructure and industrial systems with life safety implications, bringing with it significant security challenges and regulatory requirements [11].

Security and privacy thus become a serious concern, as does the flexibility and usability of the design approaches. Beyond the overarching trust management challenge, security includes data integrity, authentication, and access control at different layers of the IoT platform. Privacy means that both the content and the context around IoT data need to be protected. These requirements will be driven by various stake holders such as industry, government, consumers etc.

2.9. Communication Reliability

IoT applications can be broadly categorized into mission critical and non-mission critical. For mission critical applications, reliable communication is one of the most important features as these applications have strong QoS requirements. Reliable communication requires the following capabilities for the underlying system: (1) seamless mobility support in the face of extreme disruptions (DTN), (2) efficient routing in the presence of intermittent disconnection, (3) QoS aware routing, (4) support for redundancy at all levels of a system (device, service, network, storage etc.), and (5) support for rich communication patterns (unlike the tree-like routing structure supported by RPL developed by ROLL WG).

2.10. Self-Organization

The unified IoT platform should be able to self-organize to meet various application requirements, especially the capability to quickly discover heterogeneous and relevant (local or global) devices/data/services based on the context. This discovery can be achieved through an efficient platform-wide publish-subscribe service, or through private community grouping/clustering based upon trust and other security requirements. In the former case, the publish-subscribe service must be efficiently implemented, able to support seamless mobility, in- network caching, name-based routing, etc. In the latter case, the IoT platform needs to discover the private community groups/clusters efficiently.

Another aspect of self-organization is decoupling the sensing Infrastructure from applications. In a unified IoT platform, various applications run on top of a vast number of IoT devices. Upgrading the firmware of the IoT devices is a difficult work. It is also not practical to reprogram the IoT devices to accommodate every change of the applications. The infrastructure and the application specific logics need to be decoupled. A common interface is required to dynamically configure the interactions between the IoT devices and easily modify the application logics on top of the sensing infrastructure [23] [24].

2.11. Ad hoc and Infrastructure Mode

Depending upon whether there is communication infrastructure, an IoT system can operate either in ad-hoc or infrastructure mode.

For example, a vehicle may determine to report its location and status information to a server periodically through cellular connection, or, a group of vehicles may form an ad-hoc network that collectively detect road conditions around them. In the cases where

infrastructure is unavailable, one of the participating nodes may choose to become the temporary gateway.

The unified IoT platform needs to design a common protocol that serves both modes. Such a protocol should be able to provide: (1) energy-efficient topology discovery and data forwarding in the ad-hoc mode, and (2) scalable name resolution in the infrastructure mode.

2.12. Open API

General IoT applications involve sensing, processing, and secure content distribution occurring at various timescales and at multiple levels of hierarchy depending on the application requirements. This requires open APIs to be generic enough to support commonly used interactions between consumers, content producer, and IoT services, as opposed to proprietary APIs that are common in today's systems. Examples include pull, push, and publish/subscribe mechanisms using common naming, payload, encryption and signature schemes.

2.13. IoT Platform Management

An IoT platforms' service, control, and data plane will be governed by its own management infrastructure which includes distributed and centralized middleware, discovery, naming, self-configuring, analytic functions, and information dissemination to achieve specific IoT system objectives [18][19][20]. Towards this new IoT management mechanisms and service metrics need to be developed to measure the success of an IoTdeployment. Considering an IoT systems' defining characteristics such as, its potential large number of IoT devices, ephemeral nature to save power, mobility, and ad hoc communication, autonomic self-management mechanisms become very critical. Further considering its hierarchical information processing deployment model, the platform needs to orchestrate computational tasks according to the involved sensors and the available computation resources which may change over time. An efficient computation resource discovery and management protocol is required to facilitate this process. The trade-off between information transmission and processing is another challenge.

3. State of the Art

Over the years, many stand-alone IoT systems have been deployed in various domains. These systems usually adopt a vertical silo architecture and support a small set of pre-designated applications. A recent trend, however, is to move away from this approach, towards a unified IoT platform in which the existing silo IoT systems, as well as new systems that are rapidly deployed. This will make their data and services accessible to general Internet applications (as in

ETSI- M2M and oneM2M standards). In such a unified platform, resources can be accessed over Internet and shared across the physical boundaries of the enterprise. However, current approaches to achieve this objective are based upon Internet overlays, whose inherent inefficiencies due to IP protocol [8] hinders the platform from satisfying the IoT requirements outlined earlier (particularly in terms of scalability, security, mobility, and self-organization)

3.1. Silo IoT Architecture

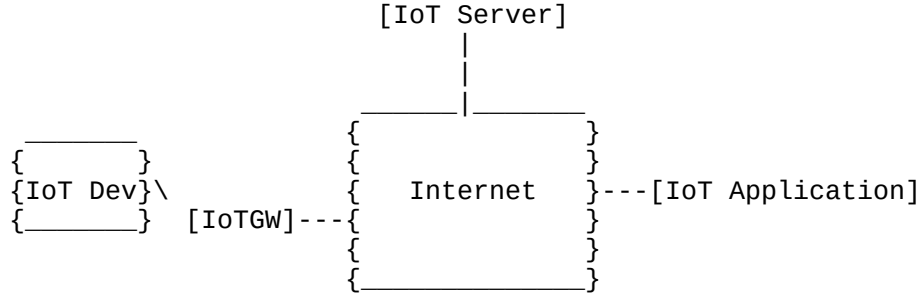


Figure 1:Silo architecture of standalone IoT systems

A typical standalone IoT system is illustrated in Figure 1, which includes devices, a gateway, a server and applications. Many IoT devices have limited power and computing resources, unable to directly run normal IP access network (Ethernet, WIFI, 3G/LTE etc.) protocols. Therefore they use the IoT gateway to the server. Through the IoT server, applications can subscribe to data collected by devices, or interact with devices.

There have been quite a few popular protocols for standalone IoT systems, such as DF-1, MelsecNet, Honeywell SDS, BACnet, etc. However, these protocols are operating at the device-level abstraction, instead of information driven, leading to a highly fragmented protocol space with limited interoperability.

3.2. Overlay Based Unified IoT Solutions

The current approach to a unified IoT platform is to make IoT gateways and servers adopt standard APIs. IoT devices connect to the Internet through the standard APIs and IoT applications subscribe and receive data through standard control and data APIs. Building on top of today’s Internet as an overlay, this is the most practical approach towards a unified IoT platform. There are ongoing

standardization efforts including ETSI[3], oneM2M[4]. Network operators can use frameworks to build common IOT gateways and servers for their customers. In addition, IETF's CORE working group [5] is developing a set of protocols like CoAP (Constrained Application Protocol) [49], that is a lightweight protocol modeled after HTTP [50] and adapted specifically for the Internet of Things (IoT). CoAP adopts the Representational State Transfer (REST) architecture with Client-Server interactions. It uses UDP as the underlying transport protocol with reliability and multicast support. Both CoAP and HTTP are considered as the suitable application level protocols for Machine-to-Machine communications, as well as IoT. For example, oneM2M (which is one of leading standards for unified M2M platform) has both the protocol bindings to HTTP and CoAP for its primitives. Figure 2 shows the architecture adopted in this approach.

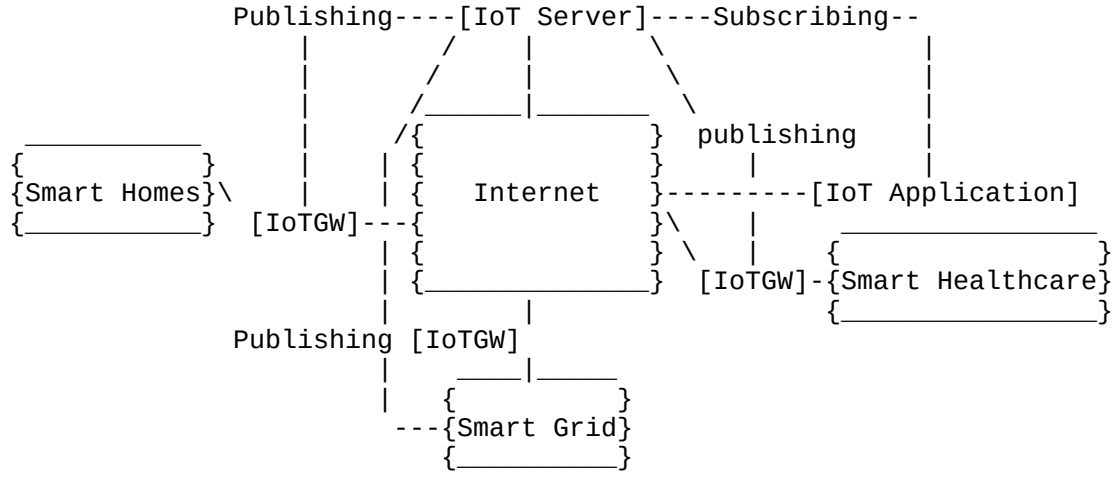


Figure 2: Implementing an open IoT platform through standarized APIs on the IoT gateways and the server

3.2.1. Weaknesses of the Overlay-based Approach

The above overlay-based approach can work with many different protocols, but the system is built upon today's IP network, which has inherent weaknesses towards supporting a unified IoT system. As a result, it cannot satisfy some of the requirements we outlined in Section 2:

- Naming. In current overlays for IoT systems the naming scheme is host centric, i.e., the name of a given resource/service is linked

to the one of device that can provide it. In turn, device names are coupled to IP addresses, which are not persistent in mobile scenarios. On the other side, in IoT systems the same service/ resource could be provided by many different devices thus requiring a different design rationale.

- Trust. Trust management schemes are still relatively weak, focusing on securing communication channels rather than managing the data that needs to be secured directly.

- Mobility. The overlay-based approach uses IP addresses as names at the network layer, which hinders the support for device/service mobility or flexible name resolution. Further the Layer 2/3 management, and application-layer addressing and forwarding required to deploy current IoT solutions limit the scalability and management of these systems.

- Resource constraints. The overlay-based approach requires every device to send data to an aggregator or to the IoT server. Resource constraints of the IoT devices, especially in power and bandwidth, could seriously limit the performance of this approach.

- Traffic Characteristics. In this approach, applications are written in a host-centric manner suitable for point-to-point communication. IoT requires multicast support that is challenging in overlay systems today.

- Contextual Communications. This overlay-based approach cannot react to dynamic contextual changes in a timely fashion. The main reason is that context lists are kept at the IoT server in this approach, and they cannot help efficiently route requests information at the network layer.

- Storage and Caching. The overlay-based approach supports application-centric storage and caching but not what ICN envisions at the network layer, or flexible storage enabled via name-based routing or name-based lookup.

- Self-Organization. The overlay-based approach is topology-based as it is bound to IP semantics, and thus does not sufficiently satisfy the self-organization requirement. In addition to topological self-organization, IoT also requires data- and service-level self-organization [59], which is not supported by the overlay approach.

- Ad-hoc and infrastructure mode. As mentioned above, the overlay-based approach lacks self-organization, and thus does not provide efficient support for the ad-hoc mode.

4. Advantages of using ICN for IoT

A key concept of ICN is the ability to name data independently from the current location at which it is stored, which simplifies caching and enables decoupling of sender and receiver. Using ICN to design an architecture for IoT data potentially provides such advantages compared to using traditional host-centric networks. This section highlights general benefits that ICN could provide to IoT networks.

- Naming of Devices, Data and Services. The heterogeneity of both network equipment deployed and services offered by IoT networks leads to a large variety of data, services and devices. While using a traditional host-centric architecture, only devices or their network interfaces are named at the network level, leaving to the application layer the task to name data and services. In many common applications of IoT networks, data and services are the main goal, and specific communication between two devices is secondary. The network distributes content and provides a service, instead of establishing a communication link between two devices. In this context, data content and services can be provided by several devices, or group of devices, hence naming data and services is often more important than naming the devices. This naming mechanism also enables self-configuration of the IoT system.

- Distributed Caching and Processing. While caching mechanisms are already used by other types of overlay networks, IoT networks can potentially benefit even more from caching and in-network processing systems, because of their resource constraints. Wireless bandwidth and power supply can be limited for multiple devices sharing a communication channel, and for small mobile devices powered by batteries. In this case, avoiding unnecessary transmissions with IoT devices to retrieve and distribute IoT data to multiple places is important, hence processing and storing such content in the network can save wireless bandwidth and battery power. Moreover, as for other types of networks, applications for IoT networks requiring shorter delays can benefit from local caches and services to reduce delays between content request and delivery.

- Decoupling between Sender and Receiver. IoT devices may be mobile and face intermittent network connectivity. When specific data is requested, such data can often be delivered by ICN without any consistent direct connectivity between devices. Apart from using structured caching systems as described previously, information can also be spread by forwarding data opportunistically.

5. ICN Challenges for IoT

This section outlines some of the ICN specific challenges [71] that must be considered when defining an IoT framework over ICN, and describes some of the trade offs that will be involved.

ICN integrates content/service/host abstraction, name-based routing, compute, caching/storage as part of the network infrastructure connecting consumers and services which meets most of the requirements discussed above; however IoT requires special considerations given heterogeneity of devices and interfaces such as for constrained networking [38][70], data processing, and content distribution models to meet specific application requirements which we identify as challenges in this section.

5.1. Naming Devices, Data, and Services

The ICN approach of named data and services (i.e., device independent naming) is typically desirable when retrieving IoT data. However, data centric naming may also pose challenges.

- Naming of devices: Naming devices is often important in an IoT network. The presence of actuators requires clients to act specifically on a device, e.g. to switch it on or off. Also, managing and monitoring the devices for administration purposes requires devices to have a specific name allowing to identify them uniquely. There are multiple ways to achieve device naming, even in systems that are data centric by nature. For example, in systems that are addressable or searchable based on metadata or sensor content, the device identifier can be included as a special kind of metadata or sensor reading.

- Size of data/service name: In information centric applications, the size of the data is typically larger than its name. For the IoT, sensors and actuators are very common, and they can generate or use data as small as a short integer containing a temperature value, or a one-byte instruction to switch off an actuator. The name of the content for each of these pieces of data has to uniquely identify the content. For this reason, many existing naming schemes have long names that are likely to be longer than the actual data content for many types of IoT applications. Furthermore, naming schemes that have self certifying properties (e.g., by creating the name based on a hash of the content), suffer from the problem that the object can only be requested when the object has been created and the content is already known, thus requiring some form of indexing service. While this is an acceptable overhead for larger data objects, it is infeasible for use when the object size is on the order of a few bytes.

- Hash-based content name: Hash algorithms are commonly used to name content in order to verify that the content is the one requested. This is only possible in contexts where the requested object is already existing, and where there is a directory service to look up names. This approach is suitable for systems with large data objects where it is important to verify the content.

- Metadata-based content name: Relying on metadata allows to generate a name for an object before it is created. However this mechanism requires metadata matching semantics.

- Naming of services: Similarly to naming of devices or data, services can be referred to with a unique identifier, provided by a specific device or by someone assigned by a central authority as the service provider. It can however also be a service provided by anyone meeting some certain metadata conditions. Example of services include content retrieval, that takes a content name/ description as input and returns the value of that content, and actuation, that takes an actuation command as input and possibly returns a status code afterwards.

- Trust: We need to ensure the name of a network element is issued by a trustworthy issuer in the context of the application, such as a trusted organization in [44]. Further the validity of each piece of data published by an authorized entity in the namespace should be verifiable - e.g., by following a hierarchical chain-of-trust to a root that is acceptable for the application. See [54]s for an example.

- Flexibility: Further challenges arise for hierarchical naming schema: referring to requirements on "constructible names" and "on-demand publishing" [28][29]. The former entails that each user is able to construct the name of a desired data item through specific algorithms and that it is possible to retrieve information also using partially specified names. The latter refers the possibility to request a content that has not yet been published in the past, thus triggering its creation.

- Control/scoping : Some information could be accessible only within a given scope. This challenge is very relevant for smart home and health monitoring applications, where privacy issues play a key role and the local scope of a home or healthcare environment may be well-defined. However, perimeter- and channel-based access control is often violated in current networks to enable over-the-wire updates and cloud-based services, so scoping is unlikely to replace a need for data-centric security in ICN.

- Confidentiality: As names can reveal information about the nature of the communication, mechanisms for name confidentiality should be available in the ICN-IoT architecture.

5.2. Name Resolution

Inter-connecting numerous IoT entities, as well as establishing reachability to them, requires a scalable name resolution system considering several dynamic factors like mobility of end points, service replication, in-network caching, failure or migration [37] [40] [41] [57]. The objective is to achieve scalable name resolution handling static and dynamic ICN entities with low complexity and control overhead. In particular, the main requirements/challenges of a name space (and the corresponding Name Resolution System where necessary) are [31] [33]:

- Scalability: The first challenge faced by ICN-IoT name resolution system is its scalability. Firstly, the approach has to support billions of objects and devices that are connected to the Internet, many of which are crossing administrative domain boundaries. Second of all, in addition to objects/devices, the name resolution system is also responsible for mapping IoT services to their network addresses. Many of these services are based upon contexts, hence dynamically changing, as pointed out in [37]. As a result, the name resolution should be able to scale gracefully to cover a large number of names/services with wide variations (e.g., hierarchical names, flat names, names with limited scope, etc.). Notice that, if hierarchical names are used, scalability can be also supported by leveraging the inherent aggregation capabilities of the hierarchy. Advanced techniques such as hyperbolic routing [53] may offer further scalability and efficiency.

- Deployability and interoperability: Graceful deployability and interoperability with existing platforms is a must to ensure a naming schema to gain success on the market [7]. As a matter of fact, besides the need to ensure coexistence between IP-centric and ICN-IoT systems, it is required to make different ICN-IoT realms, each one based on a different ICN architecture, to interoperate.

- Latency: For real-time or delay sensitive M2M application, the name resolution should not affect the overall QoS. With reference to this issue it becomes important to circumvent too centralized resolution schema (whatever the naming style, i.e, hierarchical or flat) by enforcing in-network cooperation among the different entities of the ICN-IoT system, when possible [58]. In addition, fast name lookup are necessary to ensure soft/hard real time

services [60][61][62]. This challenge is especially important for applications with stringent latency requirements, such as health monitoring, emergency handling and smart transportation [63].

- Locality and network efficiency: During name resolution the named entities closer to the consumer should be easily accessible (subject to the application requirements). This requirement is true in general because, whatever the network, if the edges are able to satisfy the requests of their consumers, the load of the core and content seek time decrease, and the overall system scalability is improved. This facet gains further relevance in those domains where an actuation on the environment has to be executed, based on the feedbacks of the ICN-IoT system, such as in robotics applications, smart grids, and industrial plants [59].

- Agility: Some data items could disappear while some other ones are created so that the name resolution system should be able to effectively take care of these dynamic conditions. In particular, this challenge applies to very dynamic scenarios (e.g., VANETs) in which data items can be tightly coupled to nodes that can appear and disappear very frequently.

5.3. Caching/Storage

In-network caching helps bring data closer to consumers, but its usage differs in constrained and infrastructure part of the IoT network. Caching in constrained networks is limited to small amounts in the order of 10KB, while caching in infrastructure part of the network can allow much larger chunks.

Caching in ICN-IoT faces several challenges:

- The main challenge is to determine which nodes on the routing path should cache the data. According to [33], caching the data on a subset of nodes can achieve a better gain than caching on every en-route routers. In particular, the authors propose a "selective caching" scheme to locate those routers with better hit probabilities to cache data. According to [34], selecting a random router to cache data is as good as caching the content everywhere. In [55], the authors suggest that edge caching provides most of the benefits of in-network caching typically discussed in NDN, with simpler deployment. However, it and other papers consider workloads that are analogous to today’s CDNs, not the IoT applications considered here. Further work is likely required to understand the appropriate caching approach for IoT applications.

- Another challenge in ICN-IoT caching is what to cache for IoT applications. In many IoT applications, customers often access a stream of sensor data, and as a result, caching a particular sensor data item may not be beneficial. In [36], the authors suggest to cache IoT services on intermediate routers, and in [37], the authors suggest to cache control information such as pub/sub lists on intermediate nodes. In addition, it is yet unclear what caching means in the context of actuation in an IoT system. For example, it could mean caching the result of a previous actuation request (using other ICN mechanisms to suppress repeated actuation requests within a given time period), or have little meaning at all if actuation uses authenticated requests as in [56].

- Another challenge is that the efficiency of Distributed Caching may be application dependent. When content popularity is heterogeneous, some content is often requested repeatedly. In that case, the network can benefit from caching. Another case where caching would be beneficial is when devices with low duty cycle are present in the network and when access to the cloud infrastructure is limited. However, using distributed caching mechanisms in the network is not useful when each object is only requested at most once, as a cache hit can only occur for the second request and later. It may also be less beneficial to have caches distributed throughout ICN nodes in cases when there are overlays of distributed repositories, e.g., a cloud or a Content Distribution Network (CDN), from which all clients can retrieve the data. Using ICN to retrieve data from such services may add some efficiency, but in case of dense occurrence of overlay CDN servers the additional benefit of caching in ICN nodes would be lower. Another example is when the name refers to an object with variable content/state. For example, when the last value for a sensor reading is requested or desired, the returned data should change every time the sensor reading is updated. In that case, ICN caching may increase the risk that cache inconsistencies result in old data being returned.

5.4. Routing and Forwarding

Routing in ICN-IoT differs from routing in traditional IP networks in that ICN routing is based upon names instead of locators. Broadly speaking, ICN routing can be categorized into the following two categories: direct name-based routing and indirect routing using a name resolution service (NRS).

- In direct name-based routing, packets are forwarded by the name of the data [57][38][42] or the name of the destination node [43]. Here, the main challenge is to keep the ICN router state required

to route/forward data low. This challenge becomes more serious when a flat naming scheme is used due to the lack of aggregation capabilities.

- In indirect routing, packets are forwarded based upon the locator of the destination node, and the locator is obtained through the name resolution service. In particular, the name-locator binding can be done either before routing (i.e., static binding) or during routing (i.e., dynamic binding). For static binding, the router state is the same as that in traditional routers, and the main challenge is the need to have fast name resolution, especially when the IoT nodes are mobile. For dynamic binding, ICN routers need to main a name-based routing table, hence the challenge of keeping the state information low. At the same time, the need of fast name resolution is also critical. Finally, another challenge is to quantify the cost associated with mobility management, especially static binding vs. dynamic binding.

During a network transaction, either the data producer or the consumer may move away and thus we need to handle the mobility to avoid information loss. ICN may differentiate mobility of a data consumer from that of a producer:

- When a consumer moves to a new location after sending out the request for Data, the Data may get lost, which requires the consumer to simply resend the request, a technique used by direct routing approach. Indirect routing approach doesn’t differentiate between consumer and producer mobility [57], also network caching can improve data recovery for this approach.

- If the data producer itself has moved, the challenge is to control the control overhead while searching for a new data producer (or for the same data producer in its new position). To this end, flooding techniques could be used, but an intra-domain level only, otherwise the network stability would be seriously impaired. For handling mobility across different domains, more sophisticated approaches could be used, including the adoption of a SDN-based control plane.

5.5. Contextual Communication

Contextualization through metadata in ICN control or application payload allows IoT applications to adapt to different environments. This enables intelligent networks which are self-configurable and enable intelligent networking among consumers and producers [36]. For example, let us look at the following smart transportation scenario: "James walks on NYC streets and wants to find an empty cab closest to his location." In this example, the context is the

relative locations of James and taxi drivers. A context service, as an IoT middleware, processes the contextual information and bridges the gap between raw sensor information and application requirements. Alternatively, naming conventions could be used to allow applications to request content in namespaces related to their local context without requiring a specific service, such as /local/geo/ mgrs/4QFJ/123/678 to retrieve objects published in the 100m grid area 4QFJ 123 678 of the military grid reference system (MGRS). In both cases, trust providers may emerge that can vouch for an application's local knowledge.

However, extracting contextual information on a real-time basis is very challenging:

- We need to have a fast context resolution service through which the involved IoT devices can continuously update its contextual information to the application (e.g., each taxi's location and Jame's information in the above example). Or, in the namespace driven approach, mechanisms for continuous nearest neighbor queries in the namespace need to be developed.

- The difficulty of this challenge grows rapidly when the number of devices involved in a context as well as the number of contexts increases.

5.6. In-network Computing

In-network computing enables ICN routers to host heterogeneous services catering to various network functions and applications needs. Contextual services for IoT networks require in-network computing, in which each sensor node or ICN router implements context reasoning [36]. Another major purpose of in-network computing is to filter and cleanse sensed data in IoT applications, that is critical as the data is noisy as is [44].

Named Function Networking [64] describes an extension of the ICN concept to named functions processed in the network, which could be used to generate data flow processing applications well-suited to, for example, time series data processing in IoT sensing applications. Related to this, is the need to support efficient function naming. Functions, input parameters, and the output result could be encapsulated in the packet header, the packet body, or mixture of the two (e.g. [24]). If functions are encapsulated in packet headers, the naming scheme affects how a computation task is routed in the network, which IoT devices are involved in the computation task (e.g. [35]), and how a name is decomposed into smaller computation tasks and deployed in the network for a better performance.

Another is challenge is related to support computing-aware routing. Normal routing is for forwarding requests to the nearest source or cache and return the data to the requester, whereas the routing for in-network computation has a different purpose. If the computation task is for aggregating sensed data, the routing strategy is to route the data to achieve a better aggregation performance [32].

In-network computing also includes synchronization challenges. Some computation tasks may need synchronizations between sub-tasks or IoT devices, e.g. a device may not send data as soon as it is available because waiting for data from the neighbours may lead to a better aggregation result; some devices may choose to sleep to save energy while waiting for the results from the neighbours; while aggregating the computation results along the path, the intermediate IoT devices may need to choose the results generated within a certain time window.

5.7. Security and Privacy

Security and privacy is crucial to all the IoT applications applications including the use cases discussed in Section 5. In one recent demonstration,it was shown that passive tire pressure sensors in cars could be hacked and used as a gateway into the automotive system [38]. The ICN paradigm is information-centric as opposed to state-of-the-art host-centric internet. Besides aspects like naming, content retrieval and caching this also has security implications. ICN advocates the model of trust in content rather than trust in network hosts. This brings in the concept of Object Security which is contrary to session-based security mechanisms such as TLS/DTLS prevalent in the current host-centric internet. Object Security is based on the idea of securing information objects unlike session-based security mechanisms which secure the communication channel between a pair of nodes. This reinforces an inherent characteristic of ICN networks i.e. to decouple senders and receivers. In the context of IoT, the Object Security model has several concrete advantages. Many IoT applications have data and services are the main goal and specific communication between two devices is secondary. Therefore, it makes more sense to secure IoT objects instead of securing the session between communicating endpoints. Though ICN includes data-centric security features the mechanisms have to be generic enough to satisfy multiplicity of policy requirements for different applications. Furthermore security and privacy concerns have to be dealt in a scenario-specific manner with respect to network function perspective spanning naming, name-resolution, routing, caching, and ICN-APIs. In general, we feel that security and privacy protection in IoT systems should mainly focus on the following aspects: confidentiality, integrity, authentication and non-repudiation, and availability.

Implementing security and privacy methods faces different challenges in the constrained and infrastructure part of the network.

- In the resource-constrained nodes, energy limitation is the biggest challenge. As an example, let us look at a typical sensor tag. Suppose the tag has a single 16-bit processor, often running at 6 MHz to save energy, with 512Bytes of RAM and 16KB of flash for program storage. Moreover, it has to deliver its data over a wireless link for at least 10,000 hours on a coin cell battery. As a result, traditional security/privacy measures are impossible to be implemented in the constrained part. In this case, one possible solution might be utilizing the physical wireless signals as security measures [46] [36].

- In the infrastructure part, we have several new threats introduced by ICN-IoT [52]:

  1. We need to ensure the name of a network element is issued by a trustworthy organization entity such as in [48], or by its trusted delegate. As name securely binds to data in ICN, security constraints of content that has not yet been published yet should also be taken into consideration.

  2. An intruder may gain access or gather information from a resource it is not entitled to. As a consequence, an adversary may examine, remove or even modify confidential information.

  3. An intruder may mimic an authorized user or network process. As a result, the intruder may forge signatures, or impersonate a source address.

  4. An adversary may manipulate the message exchange process between network entities. Such manipulation may involve replay, rerouting, mis-routing and deletion of messages.

  5. An intruder may insert fake/false sensor data into the network. The consequence might be an increase in delay and performance degradation for network services and applications.

5.8. Self-Orgnization

General IoT deployments involves heterogeneous IoT systems or subsystems within a particular scenario. Here scope-based self-organization is required to ensure logical isolation between the IoT subsystems, which should be enabled at different levels -- device/service discovery, naming, topology construction, routing over logical ICN topologies, and caching [69]. These challenges are

extended to constrained devices as well and they should be energy and device capability aware. In the infrastructure part, intelligent name-based routing, caching, in-network computing techniques should be studied to meet the scope-based self-configuration needs of ICN-IoT.

5.9. Communications Reliability

ICN offers many ingredients for reliable communication such as multi-home interest anycast over heterogeneous interfaces, caching, and forwarding intelligence for multi-path routing leveraging state-based forwarding in protocols like CCN/NDN. However these features have not been analyzed from the QoS perspective when heterogeneous traffic patterns are mixed in a router, in general QoS for ICN is an open area of research [71]. In-network reliability comes at the cost of a complex network layer; hence the research challenges here is to build redundancy and reliability in the network layer to handle a wide range of disruption scenarios such as congestion, short or long term disconnection, or last mile wireless impairments. Also an ICN network should allow features such as opportunistic store and forward mechanism to be enabled only at certain points in the network, as these mechanisms also entail overheads in the control and forwarding plane overhead which will adversely affect application throughput.

5.10. Energy Efficiency

All the optimizations for other components of the ICN-IoT system (described in earlier subsections) can lead to optimized energy efficiency. As a result, we refer the readers to read sections 5.1-5.9 for challenges associated with energy efficiency for ICN-IoT.

6. Appendix

Several types of IoT applications exists, where the goal is efficient and secure management and communication among objects in the system and with the physical world through sensors, RFIDs and other devices. Below we list a few popular IoT applications. We omit the often used term "smart", though it applies to each IoT scenario below, and posit that IoT-style interconnection of devices to make these environments "smart" in today’s terms will simply be the future norm.

6.1. Homes

The home [10] is a complex ecosystem of IoT devices and applications including climate control, home security monitoring, smoke detection, electrical metering, health/wellness, and entertainment systems. In a unified IoT platform, we would inter-connect these systems through the Internet, such that they can interact with each other and make

decisions at an aggregated level. Also, the systems can be accessed and manipulated remotely. Challenges in the home include topology independent service discovery, common protocol for heterogeneous device/application/service interaction, policy based routing/ forwarding, service mobility as well as privacy protection. Notably, the ease-of-use expectations and training of both users and installers also presents challenges in user interface and user experience design that are impacted by the complexity of network configuration, brittleness to change, configuration of trust management, etc. Finally, it is unlikely that there will be a single "home system", but rather a collection of moderately inter-operable collaborating devices. In addition, several IoT-enabled homes could form a smart district where it becomes possible to bargain resources and trade with utility suppliers.

Homes [12][13] faces the following challenges that are hard to address with IP-based overlay solutions: (1) context-aware control: home systems must make decisions (e.g., on how to control, when to collect data, where to carry out computation, when to interact with end-users, etc.) based upon the contextual information [14]; (2) inter-operability: home systems must operate with devices that adopt heterogeneous naming, trust, communication, and control systems; (3) mobility: home systems must deal with mobility caused by the movement of sensors or data receivers; (4) security: a home systems must be able to deal with foreign devices, handle a variety of user permissions (occupants of various types, guests, device manufacturers, installers and integrators, utility and infrastructure providers) and involve users in important security decisions without overwhelming them; (5) user interface / user experience: homes need to provide reasonable interfaces to their highly heterogeneous IoT networks for users with a variety of skill levels, backgrounds, cultures, interests, etc.

Smart homes have the following specific requirement for the underlying architecture:

- Smart homes require names that can enable local and wide area interactions; Also, security, privacy, and access control is particularly important for smart homes.

- Smart homes may use in-network caching at gateway to enable efficient content access.

- In smart homes, we need local, intra-domain and inter-domain routing protocols.

- In smart homes many control loops and actions depend heavily on the context, and the contexts evolve with time, e.g., temperature, weather, number of occupants, etc.

- In smart homes, local services can provide value-added contributions to a standardized home gateway network, through features such as reporting, context-based control, coordination with mobile devices, etc.

- In smart homes, the access to networked information should be shielded to protect the privacy of people, for example, cross-correlation of device activity patterns to infer higher-level activity information.

6.2. Enterprise

Enterprise building deployments, from university campuses [15] [65] [66] [67] to industrial facilities and retail complexes, drive an additional set of scalability, security, and integration requirements beyond the home, while requiring much of its ease of use and flexibility. Additionally, they bring requirements for integration with business IT systems, though often with the additional support of in-house engineering support.

Increasing number of enterprises are equipped with sensing and communication devices inside buildings, laboratories, and plants, at stadiums, in parking lots, on school buses, etc. A unified IoT platform must integrate many aspects of human interaction, H2M and M2M communication, within the enterprise, and thus enable many IoT applications that can benefit a large body of enterprise affiliates. The challenges in smart enterprise include efficient and secure device/data/resource discovery, inter-operability between different control systems, throughput scaling with number of devices, and unreliable communication due to mobility and telepresence.

Enterprises face the following challenges that are hard to address with IP-based overlay solutions: (1) efficient device/data/ resource discovery: enterprise devices must be able to quickly and securely discover requested device, data, or resources; (2) scalability: a enterprise system must be able to scale efficiently with the number and type of sensors and devices across not only a single building but multi-national corporations (for example); (3) mobility: a enterprise system must be able to deal with mobility caused by movement of devices; (4) security: security for IoT applications in the enterprise should integrate with other enterprise-wide security components.

6.3. Smart Grid

Central to the so-called "smart grid"[16] is data flow and information management, achieved by using sensors and actuators, which enables important capabilities such as substation and distribution automation. In a unified IoT platform, data collected from different smart grids can be integrated to achieve more optimizations that include reliability, real-time control, secure communications, and data privacy.

Deployment of the smart grid [17] [21] faces the following issues that are hard to address with IP-based overlay solutions: (1) scalability: future electrical grids must be able to scale gracefully to manage a large number of heterogeneous devices; (2) real time: grids must be able to perform real-time data collection, data processing and control; (3) reliability: grids must be resilient to hardware/software/networking failures; (4) security: grids and associated systems are often considered critical infrastructure -- they must be able to defend against malicious attacks, detect intrusion, and route around disruption.

Smart grids have the following specific requirements for the underlying IoT architecture:

- Smart grids require names and name resolution system that can enable networked control loops, real-time control, and security.
- Smart grids may use in-network caching to back up valuable data improving reliability.
- In smart grids, we often require very timely data delivery. Therefore, it is important to be able to locate the closest information. In addition, routing/forwarding robustness and resilience is also critical.
- In smart grids, contextual information such as location, time, voltage fluctuations, depending on the specific segment of the grid, can be used to optimize several power distribution objectives.
- In smart grids, we often rely on in-network computing to increase the scalability and efficiency of the system, putting computation closer to the data sources.
- In smart grids, energy consumptions profiles should never be disclosed at a fine granularity as it can be used to violate user privacy.

6.4. Transportation

We are currently witnessing the increasing integration of sensors into cars, other vehicles transportation systems [22]. Current production cars already carry many sensors ranging from rain gauges and accelerometers over wheel rotation/traction sensors, to cameras. These sensors can not only be used for internal vehicle functions, but they could also be networked and leveraged for applications such as monitoring external traffic/road conditions. Further, we can build vehicle-to- infrastructure (V2I),Vehicle-to-Roadside (V2R), and vehicle-to- vehicle (V2V) communications that enable many more applications for safety, convenience, entertainment, etc. The challenges for transportation include fast data/device/service discovery and association, efficient communications with mobility, trustworthy data collection and exchange.

Transportation [22][25] faces the following challenges that are hard to address with IP-based overlay solutions: (1) mobility: a transportation system must deal with a large number of mobile nodes interacting through a combination of infrastructure and ad hoc communication methods; ; also, during the journey the user might cross several realms, each one implementing different stacks (whether ICN or IP); (2) real-time and reliability: transportation systems must be able to operate in real-time and remain resilient in the presence of failures; (3) in-network computing/filtering: transportation systems will benefit from in-network computing/ filtering as such operations can reduce the end-to-end latency; (4) inter-operatibility: transportation systems must operate with heterogeneous device and protocols; (5) security: transportation systems must be resilient to malicious physical and cyber attacks.

Smart transportation applications have the following specific requirements for the underlying IoT architecture:

- Smart transportation systems require names and name resolution system to be able to handle extreme mobility, short latency and security. In addition, the mobility patterns of transportation systems increase the likelyhood that a user migrates from one network realm to another one during the journey. In this case, names and NRS should be designed in such a way to enable interoperability between different heterogeneous ICN realms and/or ICN and IP realms [68].

- Smart transportation may implement in-network caching on vehicles for efficient information dissemination

- In smart transportation, vehicle-to-vehicle ad-hoc communication is required for efficient information dissemination.

- In smart transportation, many different contexts exist, intertwined to each other and highly changing, which include location - both geographical and jurisdictional, time - absolute and relative to a schedule, traffic, speed, etc.

- In smart transportation, in-network computing is very useful to make vehicle become an active element of the system and to improve response time and scalability.

- In smart transportation, the habits of users can be inferred by looking at their movement patterns -- privacy protection is essential.

6.5. Healthcare

As more embedded medical devices, or devices that can monitor human health become increasingly deployed, healthcare is becoming a viable alternative to traditional healthcare solutions [26]. Further, consumer applications for managing and interacting with health data are a burgeoning area of research and commercial applications. For future health applications, a unified IoT platform is critical for improved patient care and consumer health support by sharing data across systems, enabling timely actuations, and lowering the time to innovation by simplifying interaction across devices from many manufacturers. Challenges in healthcare include real-time interactions, high reliability, short communication latencies, trustworthy, security and privacy, and well as defining and meeting the regulatory requirements that should impact new devices and their interconnection. In addition to this dimension, assistive robotics applications are gaining momentum to provide 24/24 7/7 assistance to patients [59].

Healthcare [26][27] faces the following challenges that are hard to address with IP-based overlay solutions: (1) real-time and reliability: healthcare systems must be able to operate on real-time and remain resilient in the presence of failures; (2) inter-operability: healthcare systems must operate with heterogeneous devices and protocols; (3) security: healthcare systems must be resilient to malicious physical and cyber attacks and meet the regulatory requirement for data security and interoperability; (4) privacy: user trust in healthcare systems is critical, and privacy considerations paramount to garner adoption and continued user; (5) user interface / user experience: the highly heterogeneous nature of real-world healthcare systems, which will continue to increase through the introduction of IoT devices, presents significant challenges in interface design that may have architectural implications.

Smart healthcare applications have the following specific requirements for the underlying IoT architecture:

- Smart healthcare system requires names and name resolution system to enable real- time interactions, dependability, and security.
- Smart healthcare may use in-network caching for rapid information dissemination.
- In smart healthcare, timely and dependable routing and information forwarding is the key.
- In smart healthcare several contexts can be used to delineate between levels of care and urgency, for example delineating between chronic, everyday, urgent, and emergency situations. Such contexts can evolve rapidly with significant impact to individuals health. Hence timely and accurate detection of contexts is critical.
- In smart healthcare, in-network computing can help resolve contexts and ensure security and dependability, as well as provide low-latency responses to urgent situations.
- In smart healthcare, personal medical data about patients should remain shielded to protect their privacy, implementing both regulatory requirements and current industry best practices.

6.6. Education

IoT technologies enable the instrumentation of a variety of environments (from greenhouses to industrial plants, homes and vehicles) to support not only their everyday operation but an understanding of how they operate -- a fundamental contribution to education. The diverse uses of hobbyist-oriented micro-controller platforms (e.g., the Arduino) and embedded systems (e.g., the Raspberry PI) point to a burgeoning community that should be supported by the next generation IoT platform because of its fundamental importance to formal and informal education.

Educational uses of IoT deployments include both learning about the operation of the system itself as well as the systems being observed and controlled. Such deployments face the following challenges that are hard to address with IP-based overlay solutions: (1) relatively simple communications patterns are obscured by many layers of translation from the host-based addressing of IP (and layer 2 configuration below) to the name-oriented interfaces provided by developers; (2) security considerations with overlay deployments and channel-based limit access to systems where read-only use of data is

not a security risk; (3) real-time communication helps make the relationship between physical phenomena and network messages easier to understand in many simple cases; (4) integration of devices from a variety of sources and manufacturers is currently quite difficult because of varying standards for basic communication, and limits experimentation; (5) programming interfaces must be carefully developed to expose important concepts clearly and in light of current best practices in education.

Smart campus systems have the following specific requirements for the underlying IoT architecture:

- Smart campus systems usually consist of heterogeneous IoT services, thus requiring names and name resolution system to enable resource/ service ownership, and be application-centric.
- Smart campus systems may use in-network caching to enable social interactions and efficient content access.
- In smart campus, inter-domain routing protocols are required which often need short latency.
- In smart campus, due to the existence of many services, relevant contextual inputs can be used to improve the quality and efficiency of different services.
- In smart campus, in-network computing services can be used to provide context for different applications.
- In smart campus, it is required to differentiate among different profiles and to allocate different rights and protection levels to them.

6.7. Entertainment, arts, and culture

IoT technologies can contribute uniquely to both the worldwide entertainment market and the fundamental human activity of creating and sharing art and culture. By supporting new types of human-computer interaction, IoT can enable new gaming, film/video, and other "content" experiences, integrating them with, for example, the lighting control of the smart home, presentation systems of the smart enterprise, or even the incentive mechanisms of smart healthcare systems (to, say, encourage and measure physical activity).

Entertainment, arts, and culture applications generate a variety of challenges for IoT: (1) notably, the ability to securely "repurpose" deployed smart systems (e.g., lighting) to create experiences; (2) low-latency communication to enable end-user responsiveness; (3)

integration with infrastructure-based sensing (e.g., computer vision) to create comprehensive interactive environments or to provide user identity information; (4) time synchronization with audio/video playback and rendering in 3D systems (5) simplicity of development and experimentation, to enable the cost- and time-efficient integration of IoT into experiences being designed without expert engineers of IoT systems; (6) security, because of integration with personal devices and smart environments, as well as billing systems.

7. Informative References

[1] Cisco System Inc., CISCO., "Cisco visual networking index: Global mobile data traffic forecast update.", 2009-2014.

[2] Shafig, M., Ji, L., Liu, A., Pang, J., and J. Wang, "A first look at cellular machine-to-machine traffic: large scale measurement and characterization.", Proceedings of the ACM Sigmetrics , 2012.

[3] The European Telecommunications Standards Institute, ETSI., "http://www.etsi.org/.", 1988.

[4] Global Intiative for M2M Standardization, oneM2M., "http://www.onem2m.org/.", 2012.

[5] Constrained RESTful Environments, CoRE., "https://datatracker.ietf.org/wg/core/charter/.", 2013.

[6] Ghodsi, A., Shenker, S., Koponen, T., Singla, A., Raghavan, B., and J. Wilcox, "Information-Centric Networking: Seeing the Forest of the Trees.", Hot Topics in Networking , 2011.

[7] Dong, L., Zhang, Y., and D. Raychaudhuri, "Enhance Content Broadcast Efficiency in Routers with Integrated Caching.", Proceedings of the IEEE Symposium on Computers and Communications (ISCC) , 2011.

[8] NSF FIA project, MobilityFirst., "http://www.nets-fia.net/", 2010.

[9] Kim, B., Lee, S., Lee, Y., Hwang, I., and Y. Rhee, "Mobiiscape: Middleware Support for Scalable Mobility Pattern Monitoring of Moving Objects in a Large-Scale City.", Journal of Systems and Software, Elsevier, 2011.

[10] Dietrich, D., Bruckne, D., Zucker, G., and P. Palensky, "Communication and Computation in Buildings: A Short Introduction and Overview", IEEE Transactions on Industrial Electronics, 2010.

[11] Keith, K., Falco, F., and K. Scarfone, "Guide to Industrial Control Systems (ICS) Security", NIST, Technical Report 800-82 Revision 1, 2013.

[12] Darianian, M. and Martin. Michael, "Smart home mobile RFID-based Internet-of-Things systems and services.", IEEE, ICACTE, 2008.

[13] Zhu, Q., Wang, R., Chen, Q., Chen, Y., and W. Qin, "IOT Gateway: Bridging Wireless Sensor Networks into Internet of Things", IEEE/IFIP, EUC, 2010.

[14] Biswas, T., Chakrabort, A., Ravindran, R., Zhang, X., and G. Wang, "Contextualized information-centric home network", ACM, Siggcomm, 2013.

[15] Huang, R., Zhang, J., Hu, Y., and J. Yang, "Smart Campus: The Developing Trends of Digital Campus", 2012.

[16] Yan, Y., Qian, Y., Hu, Y., and J. Yang, "A Survey on Smart Grid Communication Infrastructures: Motivations, Requirements and Challenges", IEEE Communications Survey and Tutorials, 2013.

[17] Miao, Y. and Y. Bu, "Research on the Architecture and Key Technology of Internet of Things (loT) Applied on Smart Grid", IEEE, ICAEE, 2010.

[18] Castro, M. and A. Jara, "An analysis of M2M platforms: challenges and opportunities for the Internet of Things", IMIS, 2012.

[19] Gubbi, J., Buyya, R., and S. Marusic, "Internet of Things (IoT): A vision, architectural elements, and future directions", Future Generation Computer Systems, 2013.

[20] Vandikas, K. and V. Tsiatsis, "Performance Evaluation of an IoT Platform. In Next Generation Mobile Apps, Services and Technologies(NGMAST)", Next Generation Mobile Apps, Services and Technologies (NGMAST), 2014.

[21] Zhang, Y., Yu, R., Nekovee, M., Liu, Y., Xie, S., and S. Gjessing, "Cognitive Machine-to-Machine Communications: Visions and Potentials for the Smart Grid", IEEE, Network, 2012.

[22] Zhou, H., Liu, B., and D. Wang, "Design and Research of Urban Intelligent Transportation System Based on the Internet of Things", Springer Link, 2012.

[23] Alessandrelli, D., Petracca, M., and P. Pagano, "T-Res: enabling reconfigurable in-network processing in IoT-based WSNs.", International Conference on Distributed Computing in Sensor Systems (DCOSS) , 2013.

[24] Kovatsch, M., Mayer, S., and B. Ostermaier, "Moving application logic from the firmware to the Cloud: towards the thin server architecture for the internet of things.", in Proc. 6th Int. Conf. on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS) , 2012.

[25] Zhang, M., Yu, T., and G. Zhai, "Smart Transport System Based on the Internet of Things", Applied Mechanics and Materials, 2012.

[26] Zhang, A., Yu, R., Nekovee, M., and S. Xie, "The Internet of Things for Ambient Assisted Living", IEEE, ITNG, 2010.

[27] Savola, R., Abie, H., and M. Sihvonen, "Towards metrics-driven adaptive security management in E-health IoT applications.", ACM, BodyNets, 2012.

[28] Jacobson, V., Smetters, D., Plass, M., Stewart, P., Thornton, J., and R. Braynard, "VoCCN: Voice-over Content-Centric Networks", ACM, ReArch, 2009.

[29] Piro, G., Cianci, I., Grieco, L., Boggia, G., and P. Camarda, "Information Centric Services in Smart Cities", ACM, Journal of Systems and Software, 2014.

[30] Ravindran, R., Biswas, T., Zhang, X., Chakrabort, A., and G. Wang, "Information-centric Networking based Homenet", IEEE/IFIP, 2013.

[31] Dannewitz, C., D’ Ambrosio, M., and V. Vercellone, "Hierarchical DHT-based name resolution for information-centric networks", 2013.

[32] Fasoloy, E., Rossiy, M., and M. Zorziy, "In-network Aggregation Techniques for Wireless Sensor Networks: A Survey", IEEE Wireless Communications, 2007.

[33] Chai, W., He, D., and I. Psaras, "Cache "less for more" in information-centric networks", ACM, IFIP, 2012.

[34] Eum, S., Nakauchi, K., Murata, M., Shoji, Yozo., and N. Nishinaga, "Catt: potential based routing with content caching for icn", IEEE Communication Magazine, 2012.

[35] Drira, W. and F. Filali, "Catt: An NDN Query Mechanism for Efficient V2X Data Collection", Eleventh Annual IEEE International Conference on Sensing, Communication, and Networking Workshops (SECON Workshops), 2014.

[36] Eum, S., Shvartzshnaider, Y., Francisco, J., Martini, R., and D. Raychaudhuri, "Enabling internet-of-things services in the mobilityfirst future internet architecture", IEEE, WoWMoM, 2012.

[37] Sun, Y., Qiao, X., Cheng, B., and J. Chen, "A low-delay, lightweight publish/subscribe architecture for delay-sensitive IOT services", IEEE, ICWS, 2013.

[38] Baccelli, E., Mehlis, C., Hahm, O., Schmidt, T., and M. Wahlisch, "Information Centric Networking in the IoT:Experiments with NDN in the Wild", ACM, ICN Siggcomm, 2014.

[39] Gronbaek, I., "Architecture for the Internet of Things (IoT): API and interconnect", IEEE, SENSORCOMM, 2008.

[40] Tian, Y., Liu, Y., Yan, Z., Wu, S., and H. Li, "RNS-A Public Resource Name Service Platform for the Internet of Things", IEEE, GreenCom, 2012.

[41] Roussos, G. and P. Chartier, "Scalable id/locator resolution for the iot", IEEE, iThings,CPSCom, 2011.

[42] Amadeo, M. and C. Campolo, "Potential of information-centric wireless sensor and actor networking", IEEE, ComManTel, 2013.

[43] Nelson, S., Bhanage, G., and D. Raychaudhuri, "GSTAR: generalized storage-aware routing for mobilityfirst in the future mobile internet", ACM, MobiArch, 2011.

[44] Trappe, W., Zhang, Y., and B. Nath, "MIAMI: methods and infrastructure for the assurance of measurement information", ACM, DMSN, 2005.

[45] Rouf, I., Mustafa, H., Taylor, T., Oh, S., Xu, W., Gruteser, M., Trappe, W., and I. Seskar, "Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study", USENIX, 2010.

[46] Liu, R. and W. Trappe, "Securing Wireless Communications at the Physical Layer", Springer, 2010.

[47] Xiao, L., Greenstein, L., Mandayam, N., and W. Trappe, "Using the physical layer for wireless authentication in time-variant channels", IEEE Transactions on Wireless Communications, 2008.

[48] Sun, S., Lannom, L., and B. Boesch, "Handle system overview", IETF, RFC3650, 2003.

[49] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <http://www.rfc-editor.org/info/rfc7252>.

[50] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <http://www.rfc-editor.org/info/rfc7230>.

[51] Sun, S., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", 2014.

[52] Liu, X., Trappe, W., and Y. Zhang, "Secure Name Resolution for Identifier-to-Locator Mappings in the Global Internet", IEEE, ICCCN, 2013.

[53] Boguna, M., Fragkiskos, P., and K. Dmitri, "Sustaining the internet with hyperbolic mapping", Nature Communications, 2010.

[54] Shang, W., "Securing building management systems using named data networking", IEEE Network 2014.

[55] Fayazbakhsh, S. and et. et al, "Less pain, most of the gain: Incrementally deployable icn", ACM, Siggcomm, 2013.

[56] Burke, J. and et. et al, "Securing instrumented environments over Content-Centric Networking: the case of lighting control", INFOCOM, Computer Communications Workshop, 2013.

[57] Li, S., Zhang, Y., Dipankar, R., and R. Ravindran, "A comparative study of MobilityFirst and NDN based ICN-IoT architectures", IEEE, QShine, 2014.

[58] Grieco, L., Alaya, M., and K. Drira, "Architecting Information Centric ETSI-M2M systems", IEEE, Pervasive and Computer Communications Workshop (PERCOM), 2014.

[59] Grieco, L., Rizzo, A., Colucci, R., Sicari, S., Piro, G., Di Paola, D., and G. Boggia, "IoT-aided robotics applications: technological implications, target domains and open issues", Computer Communications, Volume 54, 1 December, 2014.

[60] Quan, Wei., Xu, C., Guan, J., Zhang, H., and L. Grieco, "Scalable Name Lookup with Adaptive Prefix Bloom Filter for Named Data Networking", IEEE Communications Letters, 2014.

[61] Wang, Yi., Pan, T., Mi, Z., Dai, H., Guo, X., Zhang, T., Liu, B., and Q. Dong, "NameFilter: Achieving fast name lookup with low memory cost via applying two-stage Bloom filters", INFOCOM, 2013.

[62] So, W., Narayanan, A., Oran, D., and Y. Wang, "Toward fast NDN software forwarding lookup engine based on Hash tables", ACM, ANCS, 2012.

[63] Amadeo, M., Campolo, C., Iera, A., and A. Molinaro, "Named data networking for IoT: An architectural perspective", IEEE, EuCNC, 2014.

[64] Sifalakis, M., Kohler, B., Christopher, C., and C. Tschudin, "An information centric network for computing the distribution of computations", ACM, ICN Sigcomm, 2014.

[65] Lu, R., Lin, X., Zhu, H., and X. Shen, "SPARK: a new VANET-based smart parking scheme for large parking lots", INFOCOM, 2009.

[66] Wang, H. and W. He, "A reservation-based smart parking system", The First International Workshop on Cyber-Physical Networking Systems, 2011.

[67] Qian, L., "Constructing Smart Campus Based on the Cloud Computing and the Internet of Things", Computer Science 2011.

[68] Project, BonVoyage., "From Bilbao to Oslo, intermodal mobility solutions, interfaces and applications for people and goods, supported by an innovative communication network", Call H2020-MG-2014, 2015-2018.

[69] Li, S., Zhang, Y., Raychaudhuri, D., Ravindran, R., Zheng, Q., Wang, GQ., and L. Dong, "IoT Middleware over Information-Centric Network", Global Communications Conference (GLOBECOM) ICN Workshop, 2015.

[70] Li, S., Chen, J., Yu, H., Zhang, Y., Raychaudhuri, D., Ravindran, R., Gao, H., Dong, L., Wang, GQ., and H. Liu, "MF-IoT: A MobilityFirst-Based Internet of Things Architecture with Global Reachability and Communication Diversity", IEEE International Conference on Internet-of-Things Design and Implementation (IoTDI), 2016.

[71] Campolo, C., Corujo, D., Iera, A., and R. Aguiar, "Information-centric Networking for Internet-of-things: Challenges and Opportunities", IEEE Networks, Jan , 2015.

Authors' Addresses

Prof.Yanyong Zhang
WINLAB, Rutgers University
671, U.S 1
North Brunswick, NJ 08902
USA

Email: yyzhang@winlab.rutgers.edu

Prof. Dipankar Raychadhuri
WINLAB, Rutgers University
671, U.S 1
North Brunswick, NJ 08902
USA

Email: ray@winlab.rutgers.edu

Prof. Luigi Alfredo Grieco
Politecnico di Bari (DEI)
Via Orabona 4
Bari 70125
Italy

Email: alfredo.grieco@poliba.it

Prof. Emmanuel Baccelli
INRIA
Room 148, Takustrasse 9
Berlin 14195
France

Email: Emmanuel.Baccelli@inria.fr

Jeff Burke
UCLA REMAP
102 East Melnitz Hall
Los Angeles, CA 90095
USA

Email: jburke@ucla.edu

Ravishankar Ravindran
Huawei Technologies
2330 Central Expressway
Santa Clara, CA 95050
USA

Email: ravi.ravindran@huawei.com

Guoqiang Wang
Huawei Technologies
2330 Central Expressway
Santa Clara, CA 95050
USA

Email: gq.wang@huawei.com

Andres Lindgren
SICS Swedish ICT
Box 1263
Kista SE-164 29
SE

Email: andersl@sics.se

Bengt Ahlgren
SICS Swedish ICT
Box 1263
Kista, CA SE-164 29
SE

Email: bengta@sics.se

Olov Schelen
Lulea University of Technology
Lulea SE-971 87
SE

Email: lov.schelen@ltu.se