Report Number: WUCSE-2017-001

2017-07-31

Decoupling Information and Connectivity in Information-Centric Networking

Authors: Hila Ben Abraham, Jyoti Parwatikar, John Dehart, Adam Drescher, and Patrick Crowley

This paper introduces and demonstrates the concept of Information-Centric Transport as a mechanism for cleanly decoupling the information plane from the connectivity plane in Information-Centric Networking (ICN) architectures, such as NDN and CICN. These are coupled in today's incarnations of NDN and CICN through the use of forwarding strategy, which is the architectural component for deciding how to forward packets in the presence of either multiple next-hop options or dynamic feedback. As presently designed, forwarding strategy is not sustainable: application developers can only confidently specify strategy if they understand connectivity details, while network node operators can only confidently assign strategies if they understand application expectations.
We show how Information-Centric Transport allows applications to operate on the information plane, concerned only with the namespace and identities relevant to the application, leaving network node operators free to implement ICT services in whatever way makes sense for the connectivity that they manage. To illustrate ICT, we introduce sync*, a synchronization service, and show how a) its use enables applications to operate well regardless of connectivity details and... **Read complete abstract on page 2.**

Decoupling Information and Connectivity in Information-Centric Networking

Complete Abstract:

This paper introduces and demonstrates the concept of Information-Centric Transport as a mechanism for cleanly decoupling the information plane from the connectivity plane in Information-Centric Networking (ICN) architectures, such as NDN and CICN. These are coupled in today's incarnations of NDN and CICN through the use of forwarding strategy, which is the architectural component for deciding how to forward packets in the presence of either multiple next-hop options or dynamic feedback. As presently designed, forwarding strategy is not sustainable: application developers can only confidently specify strategy if they understand connectivity details, while network node operators can only confidently assign strategies if they understand application expectations.

We show how Information-Centric Transport allows applications to operate on the information plane, concerned only with the namespace and identities relevant to the application, leaving network node operators free to implement ICT services in whatever way makes sense for the connectivity that they manage. To illustrate ICT, we introduce sync*, a synchronization service, and show how a) its use enables applications to operate well regardless of connectivity details and b) its implementation can be completely managed by network operators with no knowledge of application details.

Decoupling Information and Connectivity in Information-Centric Networking

Hila Ben Abraham
Washington University in St. Louis
hila@wustl.edu

Jyoti Parwatikar
Washington University in St. Louis
jp@wustl.edu

John Dehart
Washington University in St. Louis
jdd@wustl.edu

Adam Drescher
Washington University in St. Louis
adrescher@wustl.edu

Patrick Crowley
Washington University in St. Louis
pcrowley@wustl.edu

ABSTRACT

This paper introduces and demonstrates the concept of Information-Centric Transport as a mechanism for cleanly decoupling the information plane from the connectivity plane in Information-Centric Networking (ICN) architectures, such as NDN and CICN. These are coupled in today's incarnations of NDN and CICN through the use of forwarding strategy, which is the architectural component for deciding how to forward packets in the presence of either multiple next-hop options or dynamic feedback. As presently designed, forwarding strategy is not sustainable: application developers can only confidently specify strategy if they understand connectivity details, while network node operators can only confidently assign strategies if they understand application expectations.

We show how Information-Centric Transport allows applications to operate on the information plane, concerned only with the namespace and identities relevant to the application, leaving network node operators free to implement ICT services in whatever way makes sense for the connectivity that they manage. To illustrate ICT, we introduce sync*, a synchronization service, and show how a) its use enables applications to operate well regardless of connectivity details and b) its implementation can be completely managed by network operators with no knowledge of application details.

1 INTRODUCTION

The power and promise of Information-Centric Networking (ICN) architectures, such as NDN and CICN, is based on their use of a new network abstraction for communication: the request for named data [1]. Advocates for ICN argue that the problems with IP derive from its underlying telephony-inspired abstraction, in which pairs of addressed endpoints must establish a connection to communicate (i.e, a telephone call). Hence, to properly address IP's shortcomings, one must use another abstraction.

The request for named data, an abstraction for communications that was popularized by the world-wide-web and HTTP, aims to allow applications to operate on the information plane, rather than the connectivity plane. In other words, applications should only be concerned with data namespaces and the identities of data producers and consumers, and not worry about the network characteristics. Therefore, ICNs promise to decouple applications from the details of connectivity. However, recent work [2–5] has shown that forwarding strategy, a central architectural component in NDN and CICN, binds applications to the details of connectivity in an unsustainable way.

To see why, consider that forwarding strategy is the ICN dynamic forwarding mechanism, in which a strategy determines answers to questions such as: 1) If routing rules permit multiple, equivalent next-hops, which should be chosen? , 2) If a link goes down or if a packet times out, should a packet be retransmitted on some other next hop?, and 3) Should a packet be "broadcast" to all eligible next hops?

Clearly, the answers to such questions rely on connectivity characteristics, such as where in the network the node is located (e.g, core vs. access), the number and type (e.g., wired vs. wireless) of next hop links available, and the dynamics of the network (e.g., static vs. mobile). However, these types of questions are also meaningful for application developers, because they are intrinsic to information flow and hence to application structure; for example, if an application is not certain that retransmissions will take place in the network, then the application must be structured to retransmit when necessary [5].

Presently, the two software implementations of NDN and CICN, nfd [6] and cicn [7] allow application developers to specify forwarding strategy and associate those strategy choices with namespaces. Furthermore, in current implementation, some applications require specific forwarding strategies to guarantee their correctness and performance. For instance, the new hyperbolic routing [8] scheme in NDN must use the ASF strategy in nfd to recover from local minima, and the random-per-dash-segment strategy in cicn is designed to support DASH video streams[7].

However, while a developer can pair a forwarding strategy with its application namespace in the localhost, forwarding strategies are assigned within nodes in a network by the operators of those nodes. In order to meet the expectations of applications, network node operators (the ones who apply forwarding strategies in their nodes) must therefore understand the interplay between an application's namespace and its forwarding strategy needs. In short, neither application developers nor network operators can select a forwarding strategy because the right choice depends on knowledge neither party has. This difficulty, can be mitigated in specific isolated environments where application developers also operate the entire network, such as with the global NDN testbed. However, in general, forwarding strategy unsustainably binds together application and connectivity details.

One could say that an easy solution to this problem can be that ICN services and applications should never rely on any strategy

characteristics, and should always implement their needs in the context of the application, or use a socket-based library at the end host. While we agree that strategies serve different roles — as discussed in detail in this paper — we argue that the approach of maintaining an application's logic solely in an end-to-end manner eliminates the advantages offered to applications by the adaptive, stateful, and channel-less ICN forwarding plane.

Therefore, in this work, we strive to settle this conflicted use of forwarding strategy by specifying its architectural role, and by introducing the term Information-Centric Transport (ICT). We define an ICT to be a communications service that operates solely in the information plane, dealing only with namespaces and the identities of producers and consumers, and can provide sustainable communications to application developers regardless of connectivity details. Therefore, ICT decouples the information and connectivity planes in a general-purpose way, and solves the problem of how to sustainably make forwarding strategy choices. Provided that an application developer relies on an ICT for information dissemination, then forwarding strategy choices can be made unilaterally by network node operators provided that those choices faithfully implement the ICT.

To illustrate this contribution concretely, we introduce sync*, an ICT that implements a synchronization service. We show that sync* provides reliable communications for applications under a range of connectivity scenarios, for instance when links are sufficiently intermittent to guarantee that there is never a synchronous end-to-end path available between nodes, that cause native NDN applications and traditional NDN Sync-based applications to fail.

In summary, the contributions of this paper are as follows:

- We specify the architectural role of forwarding strategies in ICN.
- We introduce the concept of Information-Centric Transport, that decouples information details from connectivity details.
- We describe and demonstrate sync*, an enhanced version of ChronoSync, as an ICT that can operate successfully in environments where there is no sustainable end-to- end path between a producer and a consumer. .

The remainder of this paper is organized as follows. The next section introduces background and related work. In Section 3, we specify the role of forwarding strategies in ICN. Section 4 introduces the concept of ICT, and Section 5 demonstrates an ICT by describing sync*. Section 6 summarizes our findings and discusses future work.

2 BACKGROUND AND RELATED WORK

2.1 ICN Architectures

Information-Centric Networking (ICN), a future internet approach, introduces a new communication model for applications and network services. While the traditional IP architecture uses addresses to identify the source and destination of every exchanged packet, the ICN approach uses names in its Interest and Data packets to request and retrieve content items. Named Data Networking (NDN) [9] and CICN [7] are two on-going architectures following the ICN approach, with two software prototypes of ICN forwarders [6, 7].

In ICN, consumers express an Interest packet to request a content by its name, and producers use Data packets when responding with the requested data. In NDN, an Interest packet is identified by the name it carries, and by a nonce value generated by the application. A nonce is used to differentiate retransmitted Interests from looped ones. ICN makes use of three data structures to forward and retrieve Interests and Data: A Forwarding Information Base (FIB) table, consists of prefixes and potential faces (upstreams) that can satisfy name requests, a Content Store (CS) that keeps a replica of a Data packet when forwarded back to the consumer, and a Pending Interest Table (PIT) that records and aggregates faces of incoming Interests to be used when a Data packet is sent back to the consumer.

When a router receives an Interest packet, it first looks for the requested Data in its CS. If there is no match for the requested name in the CS, the router searches in its PIT to check if there is a record for an unsatisfied Interest for the same name. There are two possible outcomes If a PIT entry matches the Interest's name: the router drops the Interest if it the same nonce was recorded before, and aggregates the interest if the nonce is not recognized. If no PIT entry matches the incoming Interest's name, the router searches the FIB for a list of potential upstreams that can satisfy the requested Interest. If the list of upstreams contains more than one potential face, the forwarding strategy decides to whom the Interest should be forwarded, and the router records the forwarded Interest in its PIT. When an Interest can be satisfied by a content found in a router's CS or generated by a producer, a Data packet is forwarded back to the consumer(s) by following the breadcrumbs recorded in the PIT.

Unlike IP, the FIB in ICN can contain more than one possible next hop for a namespace. In such cases, a forwarding strategy is required to decide to whom to forward the Interest. Moreover, the forwarding strategy also determines whether and when to send or retransmit an unsatisfied Interest [2, 3, 9]. Figure 1 shows the building blocks of NDN, with the strategy layer residing between the MAC layer and the Named Data layer [9].

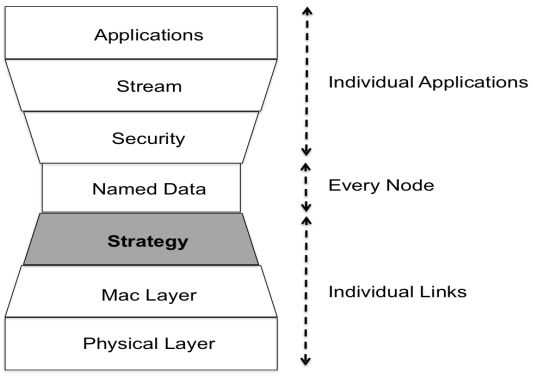

Figure 1: NDN Building Blocks as Described in [9]

2.2 Sync in ICN

The Custodian-Based Information Sharing (CBIS) system [10] was the first implementation of an ICN-based sync service. In this early paper, the authors discussed the high-level principles of what later

became the foundation of other sync protocols designed to support ICN applications.

In brief, sync can be described as the process that identifies and reconciles the set-difference of a dataset shared among multiple participants. Although in this paper we discuss sync in the context of ICN applications, it is important to note that sync's premise is widely used today by many IP-based applications, such as BitTorrent Sync, DropBox, Google Drive, and more. However, while an IP application has to design and implement its own sync paradigm to support the type of its shared data, ICN sync synchronizes namespaces that can represent any type of data. Therefore, different type of applications, from distributed file sharing[11] and chat application[12] to routing protocols[13], can be supported by the same sync service in ICN.

The following sync services for ICN have been proposed in related work: ChronoSync [14], CCNxSync [15], iSync [16], and PartialSync[17]. While these protocols can be differentiated by their implementation details, including namespace design, protocol flow, and data structures, they all follow the same high-level sync goal of providing a continuous synchronization of namespaces. In the remainder of this section we focus on describing the relevant background details of ChronoSync[14], as it is the framework we used to demonstrate the concept of ICT described in 5.

ChronoSync is a sync service library implemented in the nfd environment. To participate in the synchronization process, the distributed parties of an application first use ChronoSync API to publish a sync prefix. This sync prefix is later used by ChronoSync to notify all the registered parties about a change in their shared set of namespaces. ChronoSync uses a sync tree to represent the prefixes of the different parties, and a sequence number per party to track the changes made by it over time. A root digest is calculated to represent the current state of a particular sync tree. Two sync trees of the same sync prefix are considered up-to-date only if their root digests are equal.

Periodically, ChronoSync triggers a sync Interest containing the root digest of the local sync tree to notify remote parties about the status of the dataset. Upon recieving a sync Interest, a party compares the received digest with its local root digest to determine whether 1) Its current knowledge about of the shared set is up-to-date, 2) Its sync tree is out-of-date and missing names added by others, or, 3) Its local sync tree contains names that the Interest sender does not have.

If the root digest shows that the recipient of the sync Interest has more recent knowledge about the shared dataset, than ChronoSync responds with a Data packet to reconcile the missing names. Upon receiving of a sync Data packet, ChronoSync updates the local sync tree with the received data, and notifies the application about the new name added for the prefix. The application then can fetch the content for the new name by exchanging application-level Interest and Data packets.

2.3 Related Work

Most of related works dealing with forwarding strategies explore different mechanisms for forwarding strategies in ICN. However, our work is mainly focused on solving the problem in which the forwarding strategy component couples applications with the connectivity. In addition, while related works discussing sync explore and evaluate different sync protocols for ICN, such as ChronoSync [14], CCNxSync [15], iSync [16], our proposed sync* is an enhancement of ChronoSync with a goal to provide transport services for ChronoSync-based applications. Therefore, our contribution differ than other works proposing sync services.

Presently, nfd and cicn propose a set of forwarding strategies, each implements a set of strategy choice decisions to provide a specific forwarding behavior [7, 18]. The work in [19] discussed the principles of an adaptive forwarding strategy are discussed, while proposed strategies for such a strategy are presented in *GreenYellowRed* and [20]. A dynamic forwarding mechanism designed to discover temporary copies of content items is presented in [21]. The work in [22] proposes a revised forwarding strategy that can better prevent or detect loops in NDN.

Strategies for Wireless networks are discussed in [23, 24], and a set of adaptive forwarding strategies that for access networks in [25]. The work in [26] presents a probability-based adaptive forwarding strategy, including a statistical model to compute strategy retransmission intervals.

While the works in [3, 4] discuss application-network relation, the work in [2, 5] attempt to address the conflicts created by a specific strategy mechanism, the decision whether an Interest should be retransmitted.

The work reported in [27] proposes a consumer-producer API, built to simplify and reduce the implementation efforts of NDN application developers. Although this work can address some of the challenges considered here, we take a different approach and seek to eliminate application complexity by understanding the role of forwarding strategies and by proposing a new information-centric approach to provide application services in the network. An implementation of a socket API for ICN is discussed in [28].

3 ON THE ROLE OF FORWARDING STRATEGIES

In this section, we describe the conflicted roles the forwarding strategy presently plays, and propose a new definition to what we believe is its the correct role in the architecture.

3.1 The Problem

Studying the role forwarding strategies play in current implementations of cicn [7] and nfd [18], and in the papers described in Section 2.3, together with the name-based strategy selection software design, leads to the two contradictory assumptions. These assumptions are illustrated in Figure 2: 1) An application can choose its forwarding strategy to achieve a specific forwarding behavior, and 2) A forwarding strategy can address desired connectivity characteristics. If both assumptions are true, then a forwarding strategy couples both applications and network mechanisms, and therefore introduces challenges for an application developer who 1) cannot guarantee that the same strategy is used everywhere along the path(s) to the producer, 2) must modify its application whenever the strategy behavior is updated in future versions, and 3) should potentially develop different versions of its application to support its operation on different network connectivities. In other words,

coupling the mechanisms of both network and applications in the forwarding strategy module does not scale.

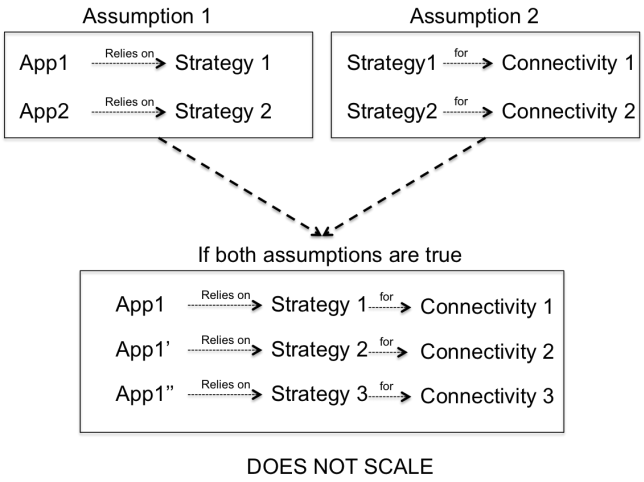

Figure 2: Strategies for Applications and Connectivity Do Not Scale

To be clear, the problem is not 'how can we distribute our forwarding strategy choice everywhere in the network' because that could be solved by including a strategy choice in the namespace forwarding information that is propagated everywhere by the routing protocol. The problem lies in an application choosing the right strategy of a given node, which depends on the specific connectivity details of that node. Indeed, some applications use one type of strategy in "core" nodes and another strategy in "access" nodes; similarly, some prefixes are shared and in fact support different applications with different forwarding strategy expectations.

We argue that a clear definition of the forwarding strategies role is an important first step towards a possible solution. Therefore, in the remainder of this section we explore the role of forwarding strategies by stepping back from the details, looking at the ICN architecture as a whole, and identifying the abstraction it aims to provide together with its way to achieve it.

3.2 From HTTP to ICN

The transition from naming endpoints in IP to naming contents in ICN not only leads to major differences in the network protocol, such as the usage of in-network caches, routers' forwarding states, and name matching rules, but also relies on a new network abstraction – the request for named data.

The concept of requesting named data is already well understood and widely used in today's Internet by the HTTP protocol. However, while both NDN and HTTP provide similar abstraction by using hierarchical names and by following the request-response communication model, they operate in fundamentally different ways.

We illustrate the abstraction provided by ICN and HTTP with the following question, asked by a consumer application: "What is the content for this name?" Here, the consumer does not specify where the content can be found, or how to get it. In theory, the consumer's question can be answered simply by broadcasting it until someone replies with the requested named data. However, broadcasting is an expensive network operation, and flooding the network is not a scalable solution. Therefore, this simple abstraction provided to applications must be translated by the network to a practical process that can efficiently find and retrieve the requested content.

In the TCP/IP model, HTTP uses names at endpoints to request the content presented by webpages. However, HTTP runs on top of the telephony network abstraction, in which IP addresses are used as the packet identifiers to establish a channel between the communicating endpoints. Therefore, in order to bridge from using names to using addresses, the TCP/IP architecture uses an additional service, the DNS protocol, to translate from the application representation of names to the network representation of addresses. The translation between the application abstraction to the network abstraction at the endpoints allows TCP/IP to provide the request for named data abstraction by the HTTP protocol. At the same time, it allows using a practical network solution that can efficiently request and retrieve HTTP packets to and from a specific destination.

In contrast to HTTP, where the name is translated to an IP address at the endpoint, the name used by an ICN application is also used by the network as the identifier of core network operations, including named-base Interest forwarding, named-based routing, and named-base caching. Therefore, ICN needs a new paradigm to translate from an application abstraction to a practical network protocol.

3.3 Information Plane Vs. Connectivity Plane

In ICN, applications ask for the data they want by name, and it is the network's job to deliver that data. But how can the network do it efficiently? In practice, there is always an actual connectivity present – e.g., a set of routers connected on top of a collection of one or more connectivity options, including WiFi links, Ethernet links, TCP channels, BT, UDP multicast, etc.. The existing connectivity must be relied upon to send and receive data. Therefore, while an ICN application operates in the *Information Plane*, the network must operate in the underlying *Connectivity Plane*.

As mentioned before, the name used in the information plane by an ICN application is also used in the connectivity plane as the identifier of network operations. Although HTTP provides similar name-based abstraction, the great benefit of using the same identifier in both the information and connectivity planes is that the application can operate without having to worry about the specific details of the connectivity plane. By contrast, IP-based applications break when 1) devices change IP addresses, 2) devices have and try to use multiple concurrent interfaces, and 3) Internet connectivity is lost.

To continue with our illustrative questions, we describe the process of moving from the information plane to the connectivity plane as answering two more questions: 1) "Who might have the data for this name?" and 2) "What is the most efficient way to retrieve it"? These two questions can and should be answered differently according to the characteristics of the network and the nature of the underlying links.

We argue that the strategy module achieves this exact goal, and bridges the gap between the information plane and the connectivity plane by answering these two questions with respect to the nature of the underlying links. Allowing a spectrum of strategies to co-exist under the umbrella of the ICN architecture provides flexible forwarding behavior that can be adapted according to the characteristics of the local connectivity. Hence, an application asks "What is the content for this name?" in the information plane, and a strategy relies on a set of input considerations in the connectivity plane when answering the questions of "Who might have the content?" and "How to retrieve it?".

While the information plane of ICN can be supported by Internet-like infrastructures in the connectivity plane by using services such as NDNS [29] and using topological design of names, it can also operate in environments where the current Internet methods do not work well, such as dynamic, non-stable topologies. Therefore, understanding how to move between the Information and connectivity planes – i.e. forwarding strategy – is a key element in the design of ICN.

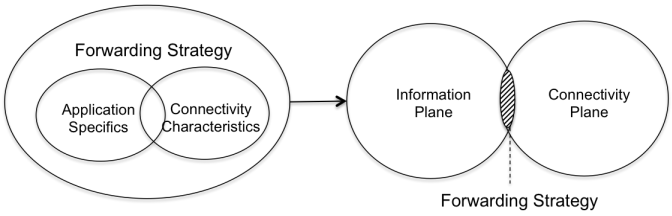

Figure 3: From coupling information and connectivity to linking information and connectivity

This section is summarized in Figure 3. We need to shift the way we think and use forwarding strategies. We argue that we need to move from them being a module that couples both applications and network mechanisms to it being an architectural component that links the information plane to the connectivity plane.

4 INFORMATION-CENTRIC TRANSPORT

Specifying the role of forwarding strategies leads to the following conclusion: applications cannot rely on forwarding strategies because these are chosen by the network operator according to the local connectivity characteristics. However, as in the IP network, some specific strategy mechanisms are required by ICN applications. For instance, as discussed in [3, 30], operations such as Interest multicasting and Interest retransmissions are currently handled by forwarding strategies and have significant impact on applications. Therefore, breaking the connection between applications and strategies eliminates the support provided to applications by such strategies.

This leads us to introduce *Information-Centric Transport (ICT)*. ICT can be viewed as an abstraction and mechanism that allows applications to operate on the information plane free from connectivity concerns. Therefore, while forwarding strategies link the information and connectivity planes, ICT serves as the interface between forwarding strategies the information plane.

To illustrate the concept of ICT, consider how it relates to traditional notions of transport. Existing transport concepts can readily be seen in the IP protocols, which can be viewed as *Connection-Centric Transport*.

- Connection-Centric Transport: concerned with endpoints and channel characteristics, such as reliability and in-order delivery.
- Information-Centric Transport: concerned with data names, the identities of information producers and consumers, and the trust relationships between identities.

In the existing IP world, transport is used to provide applications with different reliability requirements. Presently, an IP-based application can rely on existing transport protocols such as TCP or UDP, or implement its own transport mechanisms by following the Application-Level-Framing (ALF) concept[31]. Recent work has shown that similar transport frameworks [27, 28, 32] can be provided by ICN to support applications with similar end-to-end transport properties, such as real-time video conferencing and real-time video streaming [33, 34].

An ICT can be any of these cited works, but is not limited to the traditional end-to-end transport like in IP. Therefore, ICT does not eliminate the existence of such Connection-Centric transport frameworks for ICN applications, but instead, extends the concept of transport in ICN by suggesting a new type of in-network name-based transport that does not bind applications to the existence of an end-to-end path. For instance, ICT can be a socket-based library, such as the one described in [27], operating on the same machine as the application, or as an intermediate service, deployed by the network operator. When deployed in the network, an ICT should address information plane concerns while the strategy addresses concerns raised by the connectivity plane. For instance, an ICT can multicast an Interest if needed by the information plane, while multicasting due to connectivity characteristics, such as in wireless networks, is done by the forwarding strategy. In addition, Interest retransmissions due to connectivity characteristics, such as in lossy links, should be handled by the strategy, while in-network retransmissions due to information plane requirements [30], if allowed by the network operator, should be handled by the deployed ICT.

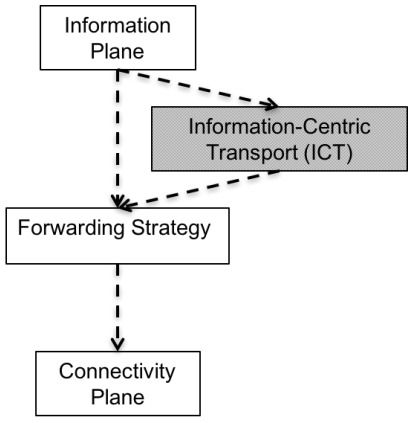

Figure 4: ICT as a potential interface between Information and Strategies

Figure 4 shows how an ICT can be used, but not required, as a mechanism between the information plane and forwarding strategies. To determine where a specific operation should be implemented, we should simply ask whether this operation is required for connectivity or information purposes. While connectivity operations should be handled by the strategy, and be completely decoupled from applications, information operations should be handled by the application and the ICT. Since the deployment of an ICT is optional and in the hands of the network operator, the application should not assume the existence of an ICT.

To conclude, we define an ICT as a service that decouples the information and connectivity planes in a general-purpose way, without knowing or relying on any application-specifics. ICT operates solely in the information plane and is placed in the network by the operator where connectivity characteristics might interrupt application functionality. Therefore it can provide sustainable communications regardless of connectivity details.

As mentioned, the goal of this paper is to specify the role of forwarding strategies in the ICN architecture, and to introduce the concept of ICT as a mechanism that addresses Information requirements presently handled by forwarding strategies. While we believe that different types of ICTs can address different sets of applications, we acknowledge that implementing and evaluating new ICTs is a long and ongoing research effort. However, to understand the potential of an ICT, this paper demonstrates sync* – an ICT built to support synchronization services in lossy environments.

5 SYNC*

This section demonstrates *sync** as an example of a secured ICT that supports sync-based applications in lossy environments. To be clear, the goal of discussing sync* is not to propose a new scalable synchronization protocol, but only to demonstrate the concept of ICT, and show how a sync-based implementation of an ICT allows ChronoSync-based [12] applications to function in environments that might have unstable links. This, without any modifications to the application, and without exploiting the network characteristics to the applications. The placement of sync* in the network is entirely up to the network operator, and applications written to use chronoSync may function with or without sync*. The advantages of sync* as an ICT for such applications is that sync* allows them to function in networks with asynchronous links, where there is never an end-to-end path between peers. Therefore, sync* decouples connectivity from information for chronoSync-based applications.

As described in Section 2.2, sync establishes data consistency over time among multiple participants. Since names can represent different types of content, sync can be used by variety of applications, such as file sharing [11, 35], chat applications [12], and routing protocols[13].

As an enhancement of ChronoSync, sync* uses the same namespace design, and the same sync tree structure to represent the prefix of the synchronized dataset. However, unlike ChronoSync, sync* operates as a separate process that can be added to any NDN router, and act as a producer of the data published under the sync tree name. As such, it can be deployed in places in the network where the operator anticipates a lossy or an unreliable communications. This way, sync* provides continuous sync service for applications regardless of the quality of the underlying links.

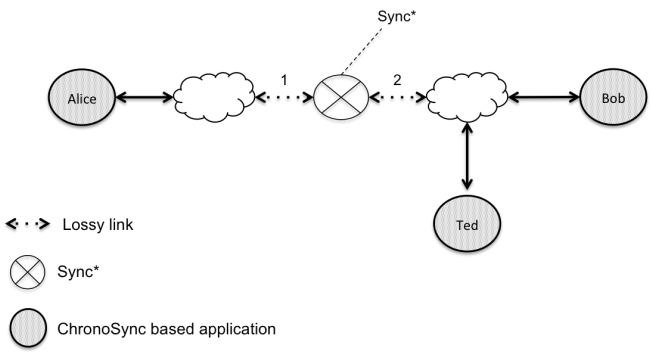

Figure 5: Sync* in Lossy Environment

Figure 5 illustrates a network where sync* can be provided to support sync applications. In the figure, Alice, Bob and Ted are participants in a ChronoSync-based application in a topology that does not always have a synchronous end-to-end path. In this specific example, if link 1 and link 2 are never up at the same time, there will never be an end-to-end path between Alice and the other participants. Therefore, the existing NDN sync services cannot synchronize Alice with any of the other participants, and Alice never gets an up-to-date view of the synchronized dataset.

This problem can be solved by using sync* in the intermediate router. Sync* responds to ChronoSync updates from participants by automatically fetching the content associated with the update. On receipt of content, it validates the signer of the content using its trust model. If the content is validated, it saves the full Data packet, including the original signature of the content and all headers. Then, sync* serves as a provider of the synchronized fetched content by registering the participant's prefix in its local NFD. Sync* can be configured to use either a persistent storage or the NDN CS to store the fetched data, depending on the characteristics of the localhost.

The characteristics of sync* as an ICT are as follows:

- The existence of Sync* does not introduce any change to the application running at the endpoints.
- The participants can use the same application in both reliable and stable networks and challenged tactical environments without introducing any new constraints to the application or the library it uses. Therefore, the application's information plane is fully decoupled from the actual connectivity plane.
- Sync* has no knowledge of the application semantics or the content it handles.
- Sync* maintains existing NDN trust schema mechanisms to fetch keys and validate the data.

While this work focuses on demonstrating an Information-Centric-Transport to support sync services, we believe that the concept of ICT can be extended to support additional classes of applications such as peer-to-peer systems like BitTorrent, and logical overlays such as DHTs. This will be further explored as part of our future work.

5.1 Demonstrating Sync*

The goal in this section is to show the capabilities of an ICT, rather than the scalability of sync*. Therefore, in these experiments, we focus on demonstrating how sync* provides a transport service for a file sharing application by decoupling it from connectivity characteristics.

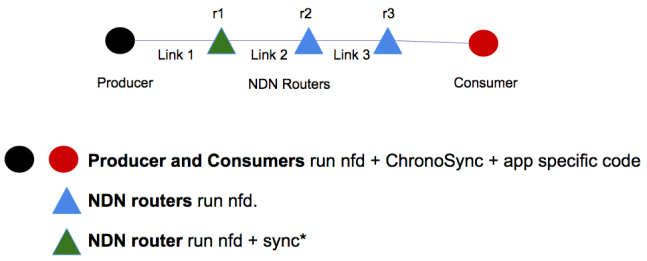

Figure 6: Tested Topology

We implemented the proposed sync* and demonstrated it in a challenged tactical network environment, where an end-to-end communication was not guaranteed. Figure 6 illustrates the topology we used, which consisted of 1 producer, 1 consumer and 3 intermediary NDN routers.

Although it seems simple, this topology illustrates an interesting use case in which a consumer and a producer communicate via a path of unreliable links and nodes, and therefore there is no consistent channel between the two. This use case can represent a sensor network, in which two sensors communicate via one or more intermediate nodes and links that can asynchronously move or fail. As we show in this section, the application completely breaks when running on top of existing communications, such as in IP-based and NDN Sync-based applications, but can successfully operate when supported by an ICT service like sync*.

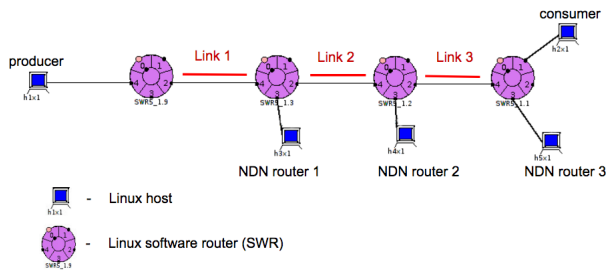

Figure 7: Mapping Topology onto Physical ONL Hardware

We conducted all our experiments on the Open Network Lab (ONL) [36], where routers and links can be programmed to control the factors we used in our experiments: link delay, link bandwidth, packet drop rates, and link availability. All machines ran nfd [6] version 0.5.1 on Ubuntu Linux (14.04.5) servers. Each experiment was repeated at least 3 times, in order to control for experimental noise caused by the dynamics of the network. Figure 7 shows the ONL setup used to create the topology in Figure 6. The producer and consumer ran ChronoSync and application-specific software. NDN routers were a combination of two machines - a Linux software router and a machine which ran the NFD code. R1 also ran sync*.

On the producer side, encrypted data was read from a local 1MB file in 1KB chunks and published by using ChronoSync API. ChronoSync then updated the sync tree to represent the sequence number of the new chunk. After the consumer and producer exchanged sync Interest and Data packets, ChronoSync on the consumer node reconciled the differences and notified the application of the new chunk name. In our experiments, we implemented the application to fetch every new chunk in order to measure its performance on top of a variety of connectivities. Once the chunk was received, the signer was validated, decrypted, and concatenated into a single file. Sync* also followed the application trust model by validating every received data packet. In our experiment, we preconfigured the nodes with the sync prefix for the content and the trust anchor needed for validation.

We used the consumer's system clock to record the start time when it received the first ChronoSync update, and the end time when it finished fetching the last chunk of the synced file. We set the Link rate to 1000Mbps, and manipulated the experimental factors to replicate an adhoc network with low bandwidth, intermittent links.

It is important to note that the measured noise in all the experiments was found to be very small relative to the measured value, and therefore, did not affect the reported results.

5.2 Similar Application, Different Communications

In this experiment we measured the average synchronization time of our ChronoSync-based file sharing application with and without sync* deployed in the network. Following the ICT concept, we used the same application when running on the different setups. When using sync*, we configured it to run on r1. In addition, we also tested the same topology with a similar IP-based application by using ipref. We recorded how many seconds it took the consumer from the moment it discovered a new chunk until it fetched the entire file.

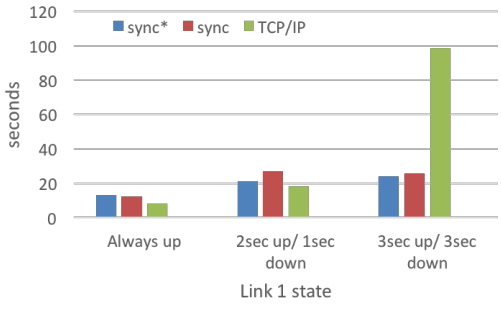

Figure 8: Application Fetch Time over Different Communications with Different Link state

Figure 8, shows the consumer fetch time on top of different communications in three different states: 1) Link 1 was alway up, hence there is a consistent end-to-end path between the consumer. 2) Link 1 was up for two seconds and down for one second. and 3) Link 1 was up for three seconds and down for three seconds. Sync* denotes the setup in which sync* ran on router 1, while sync indicates the setup in which only the end hosts ran ChronoSync.

Figure 9, shows the consumer fetch time in the scenario in which link 1 and link 2 alternated for different amount of time. Here, there was never an end-to-end path, because we stopped one link before we woke up the other. In this test the x-axis indicate the number of seconds each link was up before being stopped. The results in Figure 8 and Figure 9 support the following conclusions:

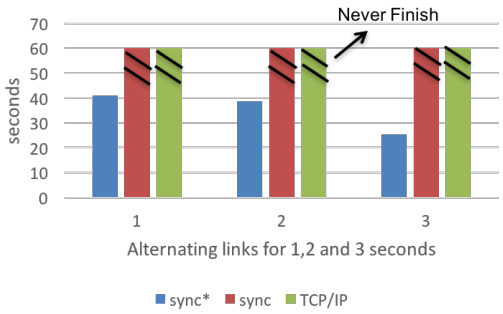

Figure 9: Application Fetch Time over Different Communications with Alternating Links (sec)

Figure 8, measured the consumer fetch time for different presents the results The first and second columns indicate the performed test. The third and forth columns show the average fetch time in seconds. When alternating links 1 and 2, there was never an end-to-end path, because we stopped one link before we woke up the other. In this test the duty cycle indicates the number of seconds each link was up before being stopped. The presented results support the following conclusions:

- The application succeeded in fetching the 1MB file when links 1 and 2 broke the end-to-end path only when sync* was running on router 1. Without sync*, the consumer failed to fetch the file
- Sync* does not improve performance when the end-to-end path is reliable and un-interrupted.

5.3 Sync* for Different Content Size

In this set of experiments, we repeated a similar scenario in which sync* ran on router1, while link 1 and link 2 alternate their up time for one, two or three seconds. As expected, the results presented in Figure 10 show a linear interaction between fetch time and the transferred data size.

5.4 Sync Vs. Sync* on Different Connectivities

In this set of experiments, we configured our application to synchronize a 1MB file on top of different connectivity characteristics. We measured the average fetch time over different connectivity characteristics, such as link delay and packet drop rate, and modify

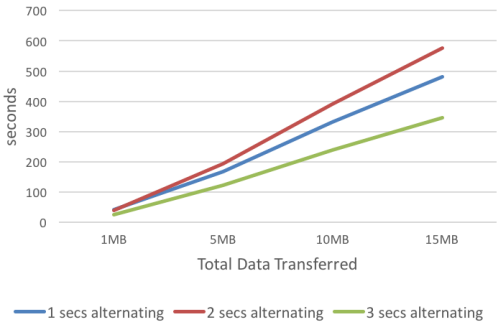

Figure 10: Application Fetch Time for Different Sizes of Data Transferred with Alternating Links (sec)

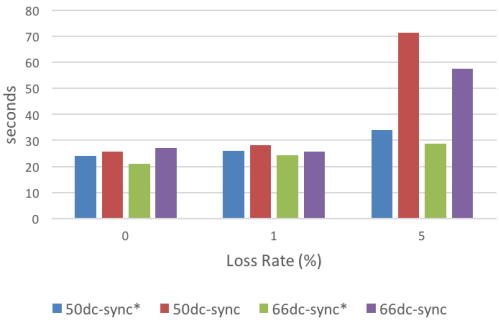

Figure 11: Application Fetch Time over Different Packet Loss Rates (%)

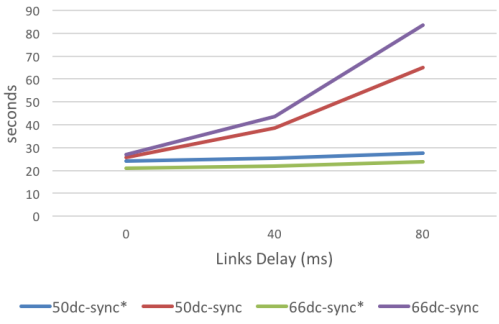

Figure 12: Application Fetch Time over Different Links Delay (ms)

the duty cycle of link 1. Duty cycle of 50% indicates that the link was up and down for three seconds each time. Duty cycle of 66% indicates that the link was up for two seconds and down for one seconds. The results are presented in figures 11 and 12.

The presented results support the following conclusions:

- With no consistent end-to-end path, running sync* on node 1 alway expedite fetch times in the tested scenarios of different link delays, and different packet loss.

- For 50% duty cycle, application can fetch up to 57% faster with sync*.
- For 66% duty cycle, application can fetch up to 72% faster with sync*.

6 CONCLUSIONS AND FUTURE WORK

In this work we proposed decoupling information and connectivity in Information-Centric Networking by specifying the role of forwarding strategies, and by introducing the concept of Information-Centric Transport (ICT). The foundation of ICT is the understanding that applications should operate in the information plane, while the network operates in the connectivity plane. Therefore, ICT fulfills the ICN abstraction – the request for named data– for applications that request to stay away from connectivity characteristics. ICT does that without obtaining any application-specific knowledge and by following the existing ICN trust mechanism. Moreover, ICT does not eliminate other applications from using traditional end-to-end transport mechanisms.

While different classes of applications might be supported by different ICTs, in this work we focus on demonstrating the ICT concept by describing sync* – a general-purpose ICT for sync services. We show that by using sync* for information dissemination, a sync-based application remains entirely in the information plane, free from any connectivity concerns. This freedom allows applications to operate in environments where IP-based applications and even present NDN sync-based applications fail.

Beyond ICN, we believe that ICT-like services, such as peer-to-peer systems like BitTorrent, logical overlays such as DHTs, and synchronization services such as NDN Sync, have been proliferating in recent years because they provide developers a mechanism for organizing distributed systems independent of connectivity details. As part of future work, we plan to explore these types of services in the context of ICT for ICN.

In addition, we plan to improve sync* as part of future work, by developing it to be a robust ICT that can efficiently scale up with the number of parallel streams it is required to support.

ACKNOWLEDGEMENTS

This work was supported by CNS-1040643 and CNS-1345282 NSF grants, and by a research gift from Cisco.

REFERENCES

[1] Van Jacobson, Diana K Smetters, James D Thornton, Michael F Plass, Nicholas H Briggs, and Rebecca L Braynard. Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pages 1–12. ACM, 2009.

[2] Hila Ben Abraham and Patrick Crowley. In-network retransmissions in named data networking. 2016.

[3] Hila Ben Abraham and Patrick Crowley. Forwarding strategies for applications in named data networking. In *Proceedings of the 2016 Symposium on Architectures for Networking and Communications Systems*, pages 111–112. ACM, 2016.

[4] Daniel Posch, Benjamin Rainer, and Hermann Hellwagner. Towards a context-aware forwarding plane in named data networking supporting qos.

[5] Hila Ben Abraham and Patrick Crowley. In-network retransmissions in named data networking. In *Proceedings of the 2016 conference on 3rd ACM Conference on Information-Centric Networking*, pages 209–210. ACM, 2016.

[6] Alexander Afanasyev et al. Nfd developer's guide. Technical report, NDN-0021, NDN, 2014.

[7] Project CICN. https://wiki.fd.io/view/Cicn.

[8] Vince Lehman, Ashlesh Gawande, Beichuan Zhang, Lixia Zhang, Rodrigo Aldecoa, Dmitri Krioukov, and Lan Wang. An experimental investigation of hyperbolic routing with a smart forwarding plane in ndn. In *Quality of Service (IWQoS), 2016 IEEE/ACM 24th International Symposium on*, pages 1–10. IEEE, 2016.

[9] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, Patrick Crowley, Christos Papadopoulos, Lan Wang, Beichuan Zhang, et al. Named data networking. *ACM SIGCOMM Computer Communication Review*, 44(3):66–73, 2014.

[10] Van Jacobson, Rebecca L Braynard, Tim Diebert, Priya Mahadevan, Marc Mosko, Nicholas H Briggs, Simon Barber, Michael F Plass, Ignacio Solis, Ersin Uzun, et al. Custodian-based information sharing. *IEEE Communications Magazine*, 50(7), 2012.

[11] Alexander Afanasyev, Zhenkai Zhu, Yingdi Yu, Lijing Wang, and Lixia Zhang. The story of chronoshare, or how ndn brought distributed secure file sharing back. In *Mobile Ad Hoc and Sensor Systems (MASS), 2015 IEEE 12th International Conference on*, pages 525–530. IEEE, 2015.

[12] Zhenkai Zhu, Chaoyi Bian, Alexander Afanasyev, Van Jacobson, and Lixia Zhang. Chronos: Serverless multi-user chat over ndn. *NDN, Technical Report NDN-0008*, 2012.

[13] AKM Hoque et al. Nlsr: Named-data link state routing protocol. In *Proceedings of the 3rd ACM SIGCOMM Workshop on Information-centric Networking*, pages 15–20. ACM, 2013.

[14] Zhenkai Zhu and Alexander Afanasyev. Let's chronosync: Decentralized dataset state synchronization in named data networking. In *ICNP*, pages 1–10, 2013.

[15] CCNx Sync. https://www.ccnx.org/releases/latest/doc/technical/SynchronizationProtocol.html.

[16] Wenliang Fu, Hila Ben Abraham, and Patrick Crowley. Synchronizing namespaces with invertible bloom filters. In *Proceedings of the Eleventh ACM/IEEE Symposium on Architectures for networking and communications systems*, pages 123–134. IEEE Computer Society, 2015.

[17] Minsheng Zhang, Vince Lehman, and Lan Wang. Partialsync: Efficient synchronization of a partial namespace in ndn. Technical report, Technical Report NDN-0039, NDN, 2016.

[18] NFD Named Data Networking Forwarding Daemon. http://named-data.net/doc/NFD/current/.

[19] Cheng Yi, Alexander Afanasyev, Lan Wang, Beichuan Zhang, and Lixia Zhang. Adaptive forwarding in named data networking.

[20] Cheng Yi et al. A case for stateful forwarding plane. *Computer Communications*, 2013.

[21] Raffaele Chiocchetti, Diego Perino, Giovanna Carofiglio, Dario Rossi, and Giuseppe Rossini. Inform: a dynamic interest forwarding mechanism for information centric networking. In *Proceedings of the 3rd ACM SIGCOMM workshop on Information-centric networking*, pages 9–14. ACM, 2013.

[22] JJ Garcia-Luna-Aceves. A fault-tolerant forwarding strategy for interest-based information centric networks. In *IFIP Networking Conference (IFIP Networking), 2015*, pages 1–9. IEEE, 2015.

[23] Giulio Grassi, Davide Pesavento, Giovanni Pau, Lixia Zhang, and Serge Fdida. Navigo: Interest forwarding by geolocations in vehicular named data networking. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2015 IEEE 16th International Symposium on a*, pages 1–10. IEEE, 2015.

[24] Marica Amadeo, Antonella Molinaro, Claudia Campolo, Manolis Sifalakis, and Christian Tschudin. Transport layer design for named data wireless networking. In *Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on*, pages 464–469. IEEE, 2014.

[25] Klaus M Schneider and Udo R Krieger. Beyond network selection: Exploiting access network heterogeneity with named data networking. In *Proceedings of the 2nd International Conference on Information-Centric Networking*, pages 137–146. ACM, 2015.

[26] Haiyang Qian, Ravishankar Ravindran, Guo-Qiang Wang, and Deep Medhi. Probability-based adaptive forwarding strategy in named data networking. In *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*, pages 1094–1101. IEEE, 2013.

[27] Ilya Moiseenko, Lijing Wang, and Lixia Zhang. Consumer/producer communication with application level framing in named data networking. In *Proceedings of the 2nd International Conference on Information-Centric Networking*, pages 99–108. ACM, 2015.

[28] Massimo Gallo, Lin Gu, Diego Perino, and Matteo Varvello. Nanet: socket api and protocol stack for process-to-content network communication. In *Proceedings of the 1st international conference on Information-centric networking*, pages 185–186. ACM, 2014.

[29] Alexander Afanasyev, Cheng Yi, Lan Wang, Beichuan Zhang, and Lixia Zhang. Scaling ndn routing: Old tale, new design. *Technical Report NDN-0004*, 2013.

[30] Hila Ben Abraham and Patrick Crowley. Controlling strategy retransmissions in named data networking. In *Proceedings of the Symposium on Architectures for Networking and Communications Systems*, pages 70–81. IEEE Press, 2017.

[31] David D Clark and David L Tennenhouse. Architectural considerations for a new generation of protocols. In *ACM SIGCOMM Computer Communication Review*,

volume 20, pages 200–208. ACM, 1990.

[32] Peter Gusev and Jeff Burke. Ndn-rtc: Real-time videoconferencing over named data networking. In *Proceedings of the 2nd International Conference on Information-Centric Networking*, pages 117–126. ACM, 2015.

[33] Lijing Wang, Ilya Moiseenko, and Lixia Zhang. Ndnlive and ndntube: Live and prerecorded video streaming over ndn. *NDN, Univ. California, Los Angeles, CA, USA, Tech. Rep. NDN-0031*, 2015.

[34] Lijing Wang, Ilya Moiseenko, and Dongsheng Wang. When video streaming meets named data networking: A case study. In *High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), 2016 IEEE 18th International Conference on*, pages 166–173. IEEE, 2016.

[35] Jared Lindblom, M Huang, Jeff Burke, and Lixia Zhang. Filesync/ndn: Peer-to-peer file sync over named data networking. *NDN, TR*, 12, 2013.

[36] Charlie Wiseman, Jonathan Turner, Michela Becchi, Patrick Crowley, John DeHart, Mart Haitjema, Shakir James, Fred Kuhns, Jing Lu, Jyoti Parwatikar, et al. A remotely accessible network processor-based router for network experimentation. In *Proceedings of the 4th ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, pages 20–29. ACM, 2008.