Journal of Industrial Information Technology and Application Vol. 2 No. 3

ISSN(Online) :2586-0852

Future Internet and Named Data Networking Hourglass, Packet and Node Architecture

Ag Asri Ag Ibrahim [1,*)], Kashif Nisar[1,*)]

[1)] Faculty of Computing and Informatics University Malaysia Sabah, Jalan UMS,
88400 Kota Kinabalu Sabah, Malaysia

Abstract. Named Data Networking (NDN) is a common network protocol for all applications and network environment. NDN's network layer protocol runs on top of a best-effort packet delivery service, which includes physical channels such as Ethernet wires, and logical connections such as UDP or TCP tunnels over the existing Internet. Using this underlying connectivity, NDN provides a content retrieval service, which allows applications to fetch uniquely named "Data packets" each carrying a piece of data. The "data" could be practically anything: text file chunks, video frames, temperature sensor readings … they are all data. Likewise, a packet in a lower layer network protocol, such as an Ethernet frame, is also a piece of data. Therefore, it should be possible to encapsulate Ethernet traffic into NDN Data packets, and establish a Virtual Private Network (VPN) through NDN communication. This post describes the architecture of a proof-of-concept Ethernet-over-NDN tunneling program, and shows a simple performance benchmark over the real world Internet.

Keywords; Future Internet, Named Data Networking, Internet of Things, Sensor Networks

1. Introduction

NDN is an entirely new architecture, but one whose design principles are derived from the successes of today's Internet, reflecting our understanding of the strengths and

[*] Corresponding author: awgasri@ums.edu.my, kashif@ums.edu.my

Received: 2017.11.27; Accepted: 2018.11.28; Published: 2018.9.30

limitations of the current Internet architecture, and one that can be rolled out through incremental deployment over the current operational Internet. The hourglass architecture is what makes the original Internet design elegant and powerful [1-3]. It centers on auniversal network layer (IP) implementing the minimal functionality necessary for global interconnectivity. This so-called "thin waist" has been a key enabler of the Internet's explosive growth, by allowing lower and upper layer technologies to innovate without unnecessary constraints. NDN keeps the same hourglass-shaped architecture as shown in Figure 1.

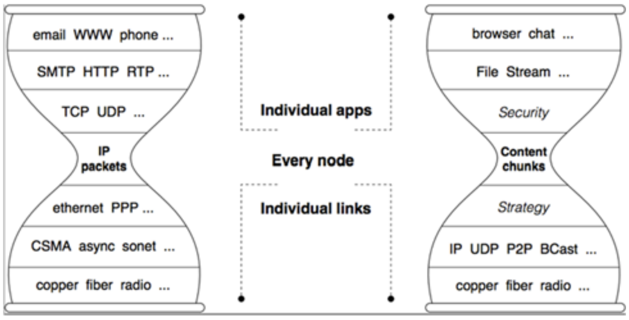

Fig. 1. Internet and NDN Hourglass Architectures

Similar to today's IP architecture, the thin waist is the centerpiece of the the NDN architecture. However, because NDN's thin waist uses data names instead of IP addresses for delivery in order to offer a new set of minimal functionality, this seemingly simple change leads to significant differences between IP and NDN in their operations of data delivery. In this section, we first give a brief sketch of the basic concepts in NDN data delivery, then explain each element and its role in the overall architecture.

NDN Packet and Node:

Each NDN packet is encoded in a Type-Length-Value (TLV) format. NDN Interest and Data packets are distinguished by the type number in the first and outmost TLV0. An NDN packet is mainly a collection of TLVs inside TLV0. Some TLVs may contain sub-TLVs, and each sub-TLV may also be further nested[4]. A guiding design principle is to keep the order of TLVis deterministic, and keep the level of nesting as small as possible to minimize both processing overhead and chances for errors.

```
LpPacket ::= LP-PACKET-TYPE TLV-LENGTH
          LpHeaderField*
          Fragment?

LpHeaderField ::= .. | Sequence
```

Sequence ::= SEQUENCE-TYPE TLV-LENGTH
fixed-width unsigned integer

Fragment ::= FRAGMENT-TYPE TLV-LENGTH
byte+

In addition, a host MUST also accept bare network packets (Interest and Data) on a NDNLPv2 link, which SHOULD be interpreted as an LpPacket with the bare network packet as its Fragment. However, such packets could be dropped later in processing if the link configured to require a certain NDNLPv2 feature but a field is missing.

LpHeaderField is a repeatable optional structure in LpHeader. NDNLPv2 features MAY add new header fields by extending the definition of LpHeaderField. Unless otherwise specified, the same field shall appear at most once. Unless otherwise specified, fields MUST appear in the order of increasing TLV-TYPE codes. If an incoming LpPacket contains an unknown LpHeaderField, the following rules apply: if the unknown field is in range [800:959], and the two least significant bits are 00, the receive SHOULD ignore the field, and continue processing the packet; otherwise, the receiver MUST drop the packet, but SHOULD NOT consider the link has an error. Note: if a field is recognized but the relevant feature is disabled, it's not an "unknown field".

Sequence contains a sequence number that is useful in multiple features. This field is REQUIRED if any enabled feature is using sequence numbers, otherwise it's OPTIONAL. Bit width of the sequence is determined on a per-link basis; 8-octet is recommended for today's links. A host MUST generate consecutive sequence numbers for outgoing packets on the same face [5].

Fragment contains a fragment of one or more network layer packets. The fragmentation and reassembly feature defines how Fragment field is constructed and interpreted. When fragmentation and reassembly feature is disabled, the Fragment field contains a whole network layer packet. Fragment is OPTIONAL; an LpPacket without Fragment is an IDLE packet.

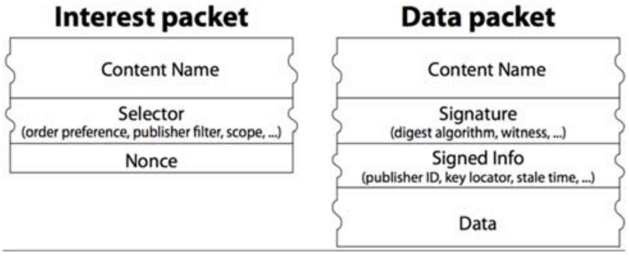
Interest packet
Content Name
Selector
(order preference, publisher filter, scope, ...)
Nonce
Data packet
Content Name
Signature
(digest algorithm, witness, ...)
Signed Info
(publisher ID, key locator, stale time, ...)
Data

Fig. 2. Internet and NDN Packet

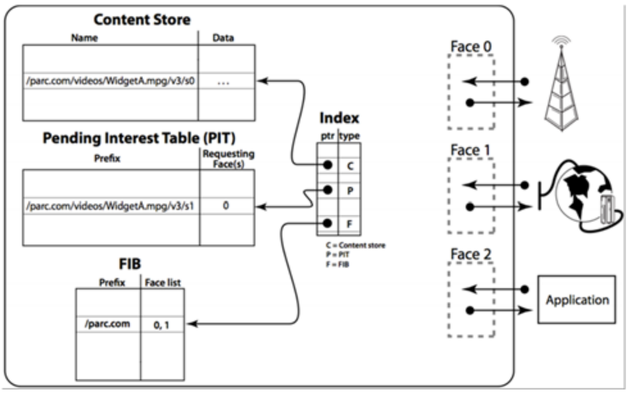

Fig. 3. Internet and NDN Node

Communication in NDN is driven by the receiving end, i.e., the data consumer. To receive data, a consumer sends out an Interest packet, which carries a name that identifies the desired data (see Figure 2). A router remembers the interface from which the request comes in, and then forwards the Interest packet by looking up the name in its Forwarding Information Base (FIB), which is populated by a name-based routing protocol. Once the Interest reaches a node that has the requested data, a Data packet is sent back, which carries both the name and the content of the data, together with a signature by the producer's key (Figure 2). This Datapacket follows in reverse the path taken by the Interest to get back to the consumer. Note that neither Interest nor Data packets carry any host or interface addresses (such as IP addresses); Interest packets are routedtowards data producers based on the names carried in the Interest packets, and Data packets are returned based on the state information set up by the Interests at each router hop (Figure 2. 3).

The router stores in a Pending Interest Table (PIT) all the Interests waiting for returning Data packets. When multiple Interests for the same data are received from downstream, only the first one is sent upstream towards the data source. Each PIT entry contains the name of the Interest and a set of interfaces from which the Interests for the same name have been received. When a Data packet arrives, the router finds the matching PIT entry and forwards the data to all the interfaces listed in the PIT entry [6]. The router then removes the corresponding PIT entry, and caches the Data in the Content Store. Because an NDN Data packet is meaningful independent of where it comes from or where it may be forwarded to, the router can cache it to satisfy future requests. Because one Data satisfies one Interest across each hop, an NDN network achieves hop-by-hop flow balance.

2. Proposed Scheme

To realize packet aggregation functions, we define the aggregated content name as

“<name prefix>/<tag>/<aggregation info>*”*

“#DC” and “#NW” are used as the <tag>. DC and NW stand for data collection and network, respectively. The “#DC” tag indicates that <aggregation info> includes the suffixes of the requested content. In addition, we define an extended table as an aggregation information table (AIT), which stores these three kinds of information. Routers wait for other Data packets and aggregate them by referencing their own AITs. For example, when a client requests content:

*“ums/parklab/*sensor1*”*

*“ums/parklab/*sensor2*”*

*“ums/parklab/*sensor3*”*

The aggregated name is

“ums/parklab/#DC/sensor1, sensor2, sensor3”

In the basic NDN, names in FIB are aggregated and recorded as prefixes. NDN routers forward Interest packets by longest-prefix matching. Therefore, the intermediate routers can forward the aggregated Interest packet using the basic NDN forwarding logic.

At a branching point, the FIB stores the information of each prefix and name. Therefore, the router can know that it must segregate the aggregated Interest packets and forward them to each content producer. When a router segregates an aggregated Interest packet, the information of <name prefix>, <tag>, and <aggregation info> is recorded in its AIT, which enables the router to aggregate the corresponding data packets. For example as shown in Fig.2, the FIB in router A stores “ums/parklab” and the face number. It can forward an aggregated interest packet named “ums/parklab/#DC/sensor1, sensor2, sensor3” with the basic NDN function.

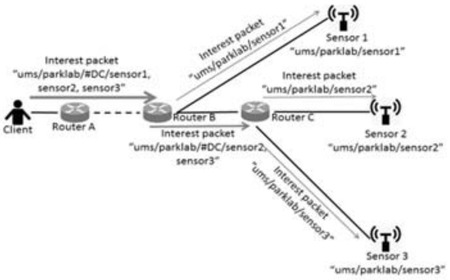

Fig. 4. Data collection with aggregated Interest packet

The FIB in router B stores two prefixes, which are “ums/parklab/sensor1” and “ums/parklab”. Router B can segregate the incoming aggregated Interest packet and create two Interest packets by referencing the FIB entries. Interest packets segregated by router B are named “ums/parklab/sensor1” and “ums/parklab/#DC/sensor2, sensor3.” At the same time, router B records the <name prefix>, <tag>, and <aggregation info> in its AIT. Router C also segregates the aggregated Interest packet and creates Interest packets named “ums/parklab/sensor2” and “ums/parklab/sensor3.” Routers B and C can wait and aggregate the corresponding Data packets by referencing their AITs. The name of the aggregated Data packet is the same as that of the incoming Interest packet. Therefore, the aggregated Data packet can also be forwarded with the basic NDN function. Routers set a timer before buffering a Data packet and the timer is managed by an efficient quality-of-service-aware waiting-time management system. When the timer expires or the router receives all requested Data packets, the router aggregates the Data packets and forwards the aggregated Data packet out. The timer is needed to avoid waiting for other Data packets for a long time when a packet loss or any other error occurs.

The “#NW” tag indicates that <aggregation info> includes the hashed value of the client’s domain. Routers associated with some hashed values record the information of Interest packets in their AITs. If a client’s application enables the corresponding Data packets to be aggregated, the application adds the tag and the hashed value to the name [9-10].

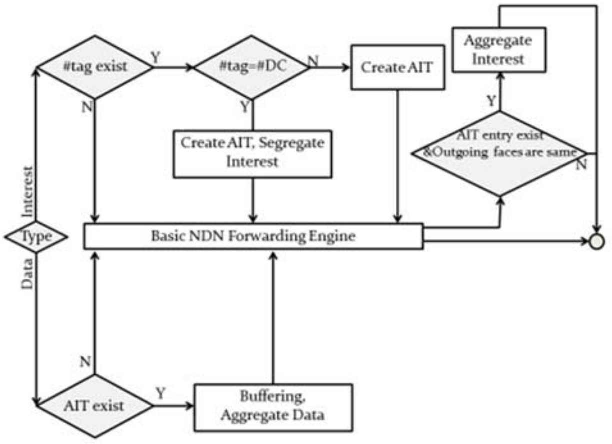
#tag exist
Y
N
#tag=#DC
N
Y
Create AIT
Aggregate Interest
Y
Create AIT, Segregate Interest
AIT entry exist &Outgoing faces are same
N
Interest
Type
Data
Basic NDN Forwarding Engine
N
AIT exist
Y
Buffering, Aggregate Data

Fig. 5. Processing logic of extended NDN router

This method does not violate the basic functions and concepts of NDN. It can be applied by adding some functions to the basic forwarding systems. Fig. 3 illustrates the processing pipeline which supports the proposed scheme. Our functions can serve as middle-ware of NDN protocol. If a tag is included in the name, the proposed functions are invoked; otherwise, the packet is forwarded by the basic NDN logic.

Traditional transport services provide point-to-point data delivery and most of today's distributed applications, including peer-to-peer applications, heavily rely on centralized servers. To aid the development of robust and efficient distributed applications, we envision a fundamentally new building block for distributed systems that we are calling Sync. Built on top of NDN's basic Interest-Data communication model, Sync utilizes naming conventions to enable multiple parties to synchronize their datasets by exchanging data digests, so that individual parties can discover and retrieve new and missing data in a most efficient and robust manner. We expect that Sync's role in the NDN architecture will evolve to one similar to TCP's in the IP architecture.

3. Results

The network performance was evaluated for ICN and CDN caching to analyze cache hits, throughput and delay. Figure 4 is shows the throughput results between ICN and CDN simulated network. Results are showing the prominent difference in improvement for throughput. The graph to calculate overall network traffic network during simulation Fig. 4 is drawn for the overall network traffic using the equation 1.

$$f(x_j) = \sum_{i=1}^{n} x_i /n \quad \textbf{(1)}$$

Where j is the time in seconds during simulation, i the events in one second throughout the network, x is the throughput in an event and n is the total number of events in one second

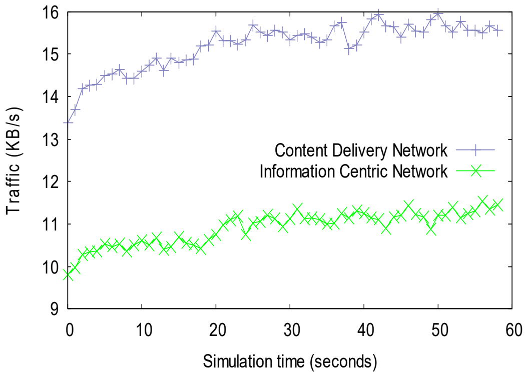

Fig. 6. Overall Network Traffic

The result in Fig. 6 is showing the cache hits while requests were sent. In a hierarchical way router nodes are able to get more data from their neighboring peers.

The data which is requested earlier by the other router nodes is cached for future requests. The graph is showing that requests for the contents which were obtain through first or second hops in network topology. The success rate of edge network cache hits compared to in-network cache was only 3% of cache hits in in-network during 60 seconds of simulation. The overall network traffic dramatically reduced by 71% of edge network cache in Fig. 4 due to in-network caching.

$$f(C_j) = \sum_{i=1}^{n} C_i \qquad \textbf{(2)}$$

Where j is the time in seconds during simulation, i the events in one second throughout the network, C is the cache hits in an event and n is the total number of events in one second.

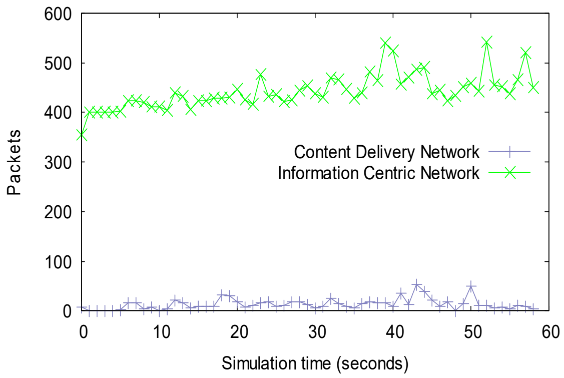

Fig. 7. The ICN and CDN Cache Hits

The graph in Fig. 7 is accentuating the overall traffic in the network was reduced when requests for contents were met by the cache in Fig. 5. This result in low traffic load at WAN communication links. The reduction of congestion in communication links leads to improve quality of network services. The removal of bottleneck in the network would provide more opportunities for other applications and services to use the same links capacity. Removal of congestion and bottleneck is result of diminishing delays. The delays sensitive applications such as real-time applications can improve tremendously.

4. Conclusions

Mobility is one of the most significant issues in NDN. Moreover, mobility functions are moved from the mobile devices to the network, and the BIT is easily maintained to keep the table size limited. The implications of ICN platform are highly beneficial for different industrial sectors. The information centric network does not only offer cache mechanism to boost the access but also provide content encryption instead of securing the communication link. The security of content itself will also reduce the conventional security devices cost.

References

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, “A survey of Information-Centric Networking,” IEEE Communications Magazine, Volume 50, Issue 7, pp. 26-36, July 2012.

[2] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, et al., “A survey of Information-Centric Networking research,” IEEE Communications Surveys & Tutorials, Volume 16, Issue 2, pp. 1024-1049, July 2013.

[3] L. Zhang, V. Jacobson, B. Zhang, G. Tsudik, K. Claffy, et al., “Named data networking (NDN) project,” NDN Technical Report NDN-0001, October 2010.

[4] L. Zhang, K. Claffy, P. Crowley, C. Papadopoulos, L. Wang, et al., “Named data networking,” NDN Technical Report NDN-0019, April 2014.

[5] D. Byun, B.J. Lee, and M.-W. Jang, “Adaptive flow control via interest aggregation in CCN,” IEEE International Conference on Communications (ICC), pp. 3738-3742, June 2013.

[6] S. Harada, Z. Yan, Y.J. Park, and W. Kameyama, “Packet aggregation and segregation mechanism over named data networking,” IEICE Technical Report, Volume 114, Number 252, NS2014-121, pp. 99-103, October, 2014.

[7] Sho Harada, Zhiwei Yan, Yong-Jin Park, and Wataru Kameyama, “Performance analysis of packet aggregation over NDN”, IEICE General Conference, BS-3-29, March, 2015.