Seamless Producer Mobility as a Service in Information Centric Networks

Aytac Azgin, Ravishankar Ravindran, Asit Chakraborti, and G.Q. Wang
Huawei Research Center, Santa Clara, CA, USA.
{aytac.azgin, ravi.ravindran, asit.chakraborti, gq.wang}@huawei.com

ABSTRACT

In this paper, we present a service-driven mobility support architecture for Information Centric Networks that provides seamless mobility as an on-demand network service, which can be enabled/disabled based on network capabilities or resource availability. Proposed architecture relies on the ID/Locator split on ICN namespaces to support the use of persistent names and avoids name reconfiguration due to mobility. We implemented the proposed solution over a service-centric CCN platform, with multiple end-hosts running a video conferencing application acting as Consumers and Producers, and observed its capability to support seamless handover.

CCS Concepts

•**Networks** → **Network architectures; Network protocols;** *Network performance evaluation;*

Keywords

Information-centric networks, content-centric networking, named data networking, producer mobility, mobility as a service, locator/ID split

1. INTRODUCTION

Information-centric Networking (ICN) addresses the shortcomings of current Internet architecture by moving away from host-centric communication model towards a new content-centric one, where the named content becomes the principal entity for information dissemination [12]. In this paper, we focus on one such architecture, the content-centric networking architecture (*i.e.*, CCN/NDN), and propose a service-centric solution to seamlessly handle content mobility in CCN.

CCN/NDN based architectures utilize hierarchically structured names and a pull based approach to content delivery (by sending Interests to retrieve Data), while ensuring stateful forwarding with the use of pending Interest tables (PITs) to store information on the forwarded requests. Content authenticity is provided through the digital signatures carried within Data packets. Despite the flexible support CCN/NDN based architectures provide to retrieve content from anywhere in the network, such an approach by itself becomes insufficient to handle content delivery from mobile hosts (Producer mobility) for various reasons.[1] For instance, the overhead associated with re-routing Interests towards mobile hosts can be overwhelming, and such on-demand resolution after handovers can introduce unpredictable delays further causing performance issues at the end hosts [5].

1.1 How to Handle Producer Mobility in CCN/NDN?

To address Producer mobility in CCN/NDN based architectures, various approaches have been proposed that can be characterized as host-driven (or application-driven within CCN context) or network-driven solutions. In a host-driven approach, end host is responsible for announcing its reachability to the network and triggering related changes in network state to enable routing continuity [13, 4]. For instance, in [13], the authors propose an anchor-based solution, where the end hosts agree on an anchor point and use application initiated requests to handle Producer mobility. Here, the solution utilizes the existing PIT to store information on these requests to guide Interests and Data packets towards first the anchor then the client application. As the approach uses application-specific (and pre-defined) anchors, it can introduce significant path stretch. Furthermore, it may require significant overhead to set up and maintain the traces, especially with high levels of mobility and increased number of namespaces associated with mobile hosts. Furthermore, the use of publicly accessible PIT entries can potentially create security risks (*e.g.*, flooding attacks by malicious applications).

In [4] the authors propose an anchorless solution, which utilizes a temporary forwarding information base (TFIB, which can be implemented within a router's FIB) to create temporary entries that are prioritized by the Consumer Interests to discover the outgoing interface. Specifically, after a handover, Producer sends to itself (*i.e.*, location indicated by the FIB entries) a specific type of Interest, which helps setup TFIB entries along the path. In doing so, future Interests targeting the Producer can be routed (or re-directed) towards its current location. As the solution relies on existing routing protocols to converge to the correct path after handovers, depending on mobility levels and frequency of such updates, path stretch can increase to unacceptable levels.

[1]Consumer mobility is inherently supported, as the Consumer can send fresh requests after changing its point of attachment.

ICN'16, September 26-28, 2016, Kyoto, Japan

DOI: http://dx.doi.org/10.1145/2984356.2988521

Furthermore, depending on the path stretch, signaling overhead can become a serious concern. Lastly, as the solution relies on end hosts or applications modifying FIB entries (assuming the architecture uses the existing FIB to store TFIB entries), it has to be supported by all the service routers at each domain hosting a mobile Producer (which may not be possible due to security concerns associated with such action).

In [6] we proposed a network-driven approach, where the reachability of the mobile host is handled by the network, by using the late-binding technique (which is also considered in [3] to scale routing). Applications can explicitly request mobility support for a given namespace during its registration, with the host mobility tracked by the network through the distributed controller nodes. As the Producer moves, controllers are updated with the most recent locator information (local controllers for intra-domain mobility, and home controllers for inter-domain mobility), which is then used to deliver Consumer Interests. As the solution employs locator/ID split, it can avoid routing churn caused by Producer mobility, while offering better scalability as the number of Producers or mobile namespaces increases (due to limited scope for the updates). Our solution, as we explain next, also considers the use of locator/ID split to manage the namespace mobility.

1.2 Why Locator/ID Split? And Its Use in CCN/NDN

We consider locator/ID split as an important requirement to handle mobility at the network layer [10], which can inherently be supported by the ICN architectures (such as MobilityFirst [11]) due to unique names assigned to each entity and routing on names to resolve locations, and achieve seamless connectivity. Note that, such objectives cannot be achieved efficiently with the current IP through the use of protocols based on Mobile IP, due to the use of IP address as a locator and as an identifier, triangular routing and the control overhead [14].

In CCN/NDN based architectures, using hierarchical identifiers for routing cannot scale due to potentially explosive growth in namespaces and the diminishing of aggregatability with content mobility, multi-homing, or resource replication [2]. Furthermore, practical problems such as name-suffix hole may arise when names are used for network reachability. Routing scalability is typically achieved by designing names with aggregatable property, which is the case for IP today. However, having such feature in CCN/NDN would lead to relinquishing the persistency of names, as the names would involve a topological component for scalability (which also suggests resources to be renamed depending on, for instance, network or business specs or characteristics). Furthermore, overloading an identifier as a locator can lead to unstable routing control and forwarding plane operations.

Locator/ID split in CCN/NDN therefore imposes splitting the hierarchical namespace to support routable, persistent and human-friendly names. In such case, names would be divided based on application binding vs. advertised network entities (in routing plane) to achieve scalable routing. For instance, a persistent identifier /Content-Provider/Content-Type/Content-Name, which would be used to create secure content objects, can be published by multiple content distributers, where it would be mapped to different locators, such as /Content-Distributor/Region/Zone/Storage, to resolve the content names to specific infrastructure entities. The fundamental requirement with this form of splitting is no different than that of MobilityFirst or LISP [7], which is the requirement of a name resolution system (NRS) to map the two namespaces. Even though the design of such NRS is not the focus of this research, many factors due to human readability can be considered in its design: (*i*) contextual nature of hierarchical names that allows mapping names to authoritative domain, which can realize a decentralized NRS (compared to the flat architecture required by MobilityFirst), or (*ii*) the ability to route using both ID and Locator based on network domain/segment, or the nature of resource entity being fixed or mobile, or the routing/forwarwding scalability requirements at the network layer.

1.3 Contributions

In this paper, we extend our earlier approach in [6] to realize it as a service-centric mobility support solution for ICN (with specific emphasis on CCN/NDN [2]) that is capable of accommodating both host-driven and make-before-break based network-driven mobility mechanisms.[3] Proposed solution can achieve scalable performance for both intra- and inter-domain mobility with resource efficient control/data plane operations. Our solution can support persistent names from an application's perspective, thereby avoiding name reconfiguration due to mobility. As the mobility is offered as a service, proposed solution enables a Producer to explicitly seek mobility support for an application prefix instead of network providing mobility support for all Interest flows, and allows mobility support to be enabled/disabled based on resource availability/requirements or network capabilities. As mobility states are quickly updated, we can minimize packet loss due to host mobility.

The rest of the paper is organized as follows. In **Section** 2, we introduce the proposed mobility support architecture, and explain its components and their use pre-, during, and post- handover. We evaluate the performance of the proposed architecture in **Section** 3 over a CCN-based service platform that implements a video conferencing application between static and mobile end hosts. **Section** 4 concludes our paper.

2. SERVICE DRIVEN MOBILITY ARCHITECTURE

Figure 1 illustrates the proposed service-driven mobility support architecture for ICN, which is motivated by a top-down model considered for 5G Next Generation Mobile Networks (NGMN) [1]. 5G NGMN framework allows realization of new architectures as on-demand *service slice*s that assume an end-to-end programmable infrastructure spanning user entity, edge cloud, core network, and data centers with compute, storage, bandwidth resources. **ICN can be one such slice to support the ICN services**. For services over ICN, mobility can be handled in its own slice at the network layer (unlike the complex and inefficient anchor-based solutions offered for LTE networks). Proposed solution allows mobility as an on-demand service, which is enabled for services

[2]In the remainder of the paper, we will use the terms ICN and CCN interchangeably, as our research focuses on a CCN/NDN based ICN architecture.

[3]Due to space limitations, hereafter we only discuss mechanisms related to network-driven mobility.

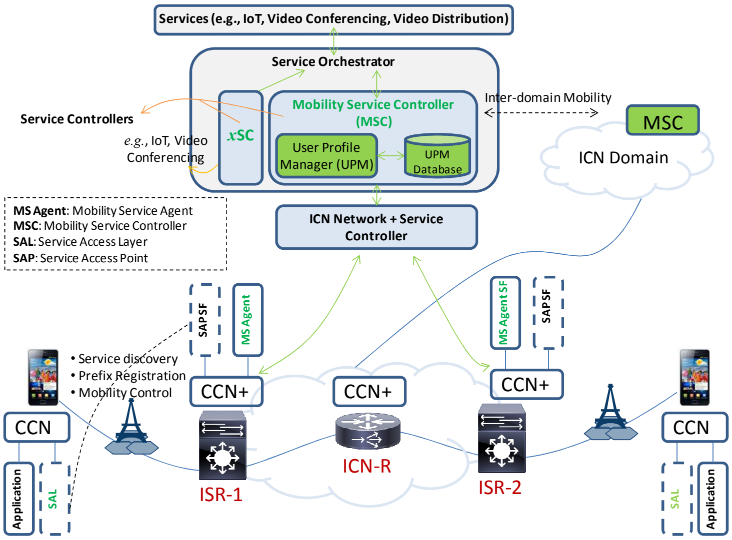

Figure 1: Architectural components to support ICN mobility.

that request for it.

The main components for the proposed architecture are explained as follows (from top to bottom):

- **Service Orchestrator** layer (`SOL`) exposes application programming interfaces (`APIs`) to heterogenous services to request the desired service with meta information (*e.g.*, geographical distribution of demand and/or performance). Additionally, `SOL` provides several handles for managing and monitoring of the services. For instance, as shown in Figure 1, `SOL` interfaces with IoT, video conferencing and video distribution services, each of which carries different requirements.
  `SOL` hosts several service specific orchestrators that determine the placement of network and service functions, along with their connectivity requirements based on the expected service load.

- **Service Controllers** (`SCs`) translate requested services to provisionable resources, and pass them to the **ICN Network Controller**.

- **Mobility Service Controller** (`MSC`) is the `SC` that is responsible for managing the service profiles and resolving the mapping of names for the requests received from the `ICN` infrastructure to locators (e.g., /`ServiceNode`$_X$). For instance, service profiles may include the mobility service specs and their service level agreements (`SLAs`) (*e.g.*, geography limits for providing the mobility service). Within `MSC`, the entity that maps services and names with the associated mobility requirements is referred to as the **User Profile Manager**. With the help of `MSC`, we can enable *mobility-as-a-service*, and allow content providers to request to have mobility service enabled for their subscribers. In such case, `MSC` would push policies into the `ICN` infrastructure to provide mobility support for the corresponding service flows.
  NOTE: Proposed architecture assumes a *decentralized controller framework to name resolution*, with distributed `MSCs` in different domains coordinating to resolve the location of mobile hosts (and their namespaces) part of the mobility service. To facilitate fast discovery, namespaces can be constructed as DomainId::HostId ::ContentId, with the DomainId component used to identify the home `MSC` for a given host.

- **ICN Service Controller** manages the compute virtualization of `ICN` infrastructure by monitoring the available compute capacity within and satisfying the requests from `SOL` to provision the service functions in specific `ICN` routers (*i.e.*, `ICN` Service Router or `ISR`).

- **ICN Network Controller** (`INC`) manages the network virtualization of `ICN` infrastructure by hosting service specific network controllers including `MSC`. As an example, `INC` is responsible for dynamically provisioning the system resources (*e.g.*, forwarding tables) based on the service requirements.

Packet forwarding is handled by two types of `ICN` routers, **ICN Service Routers** (`ISRs`, which reside at the network edges) and **ICN Relay** (`ICN-R`) nodes (which act as overlays between `ISRs`). Service functions (such as for mobility [4]) are installed at the `ISRs` to help with several edge services. **Mobility Service Agent** (`MS-Agent`) at the `ISRs`, represents the mobility-related service function that is managed by the `MSC`. `MS-Agents` are responsible for (*i*) (de-)registration of service names requiring mobility support and (*ii*) resolving content (or entity) names to locators by communicating with the `MSC`. **Service Access Point** (`SAP`) at the `ISR` represents the service function responsible for the discovery of mobility service in the network and with helping to identify `APIs` to (de-)register for mobility service. **Service Access Layer** (`SAL`) at the `UE` acts as a proxy to handle mobility service signaling for the application (*e.g.*, requesting mobility service for a name prefix). [5]

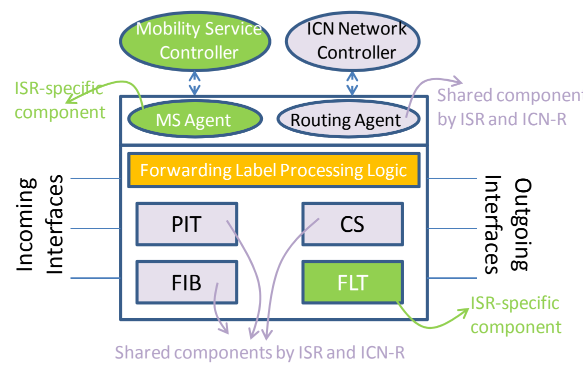

Figure 2: General architecture for the types of ICN/CCN routers.

We illustrate the general architecture for `ICN` routers in Figure 2. Differences in regards to mobility service control plane is explained above (*i.e.*, `MS-Agent` agent is located only at the `ISR`). Both routers share the same basic components

[4]Service functions are virtual machines or containers executing a specific service logic as part of the global service execution.

[5]ISR functions that handle mobility can also be part of the point of attachment, base station or access point.

of **Forwarding Information Base** (FIB), **Pending Interest Table** (PIT), and **Content Store** (CS). To enable mobility service support at the data plane, instead of relying solely on the FIB, we introduce a new data structure at the ISRs, referred to as **Forwarding Label Cache Table** or FLT to cache the name-to-locator mappings (*i.e.*, forwarding labels). FLT is implemented as a hash-table (with longest-prefix matching used to locate a matching entry), and its entries are controlled by the MSC through on-demand resolution or proactive provisioning (suggesting, entries expire after a certain timeout on a scale of minutes).[6] After an FLT entry is deleted due to timeout, any incoming request triggers a new resolution through MSC. Also note that, FLT is a software-defined component, so it can be used by any other service to provide opportunistic routing to the supported services.

2.1 Forwarding Label

To realize ID/locator split in CCN, in a recent ICNRG Internet draft [9], we proposed a Forwarding Label (FL) object that acts as a locator and provides the flexibility to forward Interests on a name other than the one provided within the original Interest, while allowing the ability to modify it on-the-fly. Proposed FL-object helps with not only mobility but also with opportunistic routing, binding Interests to services at a given location, and in-network computing, thereby allowing for incremental enhancement over CCN to provide richer services at the network edge.

FL objects are considered as container objects that include Locator ID (LID), service specific metadata (*i.e.*, contextual information on the application/service to help the network triggering appropriate FL processing such as trust validation) and (optional) security attributes for authentication. LID is considered as a hierarchically structured topological name representing domain, gateway, or host IDs.

FL objects are inserted within the Interest either by the consuming application (which may require a trust binding between the ID and the LID) or by the network (which typically occur at the ingress service routers, if the Interest satisfies an existing flow service profile). As an FL object can be modified within the network (*e.g.*, at domain boundaries, network edges, etc.), it is considered as part of the optional hop-by-hop header. In regards to processing of FL-object carrying Interests, various options are available depending on service profiles and trust relationships, *e.g.*, LID preference (over ID), ID preference (over LID), LID preference with swap, or LID ignore.

We refer the reader to [9] for more detailed information on its format and use.

2.2 Packet Forwarding Logic

We illustrate the forwarding logic for Interest processing in Figure 3 (regular CCN processing is used for Data packets), where the green colored boxes represent the new states used for *mobility-service* (MS) enabled request processing. Interest flows that invoke mobility support can be identified using one of the following approaches: (*i*) by setting the **mobility-service flag** within the Interest (*e.g.*, by the application at the UE), (*ii*) by prepending the content name with the /Mobility-Service tag, or (*iii*) by provisioning traffic policy rules at the ISR to monitor for Interest flows with pre-configured names matching a prefix of the Interest (or by checking the Interest's attributes to distinguish flows requiring seamless mobility support). If mobility-service is not enabled for a received Interest, regular CCN processing logic is used, through *PIT processing*, *CS processing*, and *FIB processing*. On the other hand, if the mobility-service is enabled, then the proposed FL processing logic (FLPL) is used.

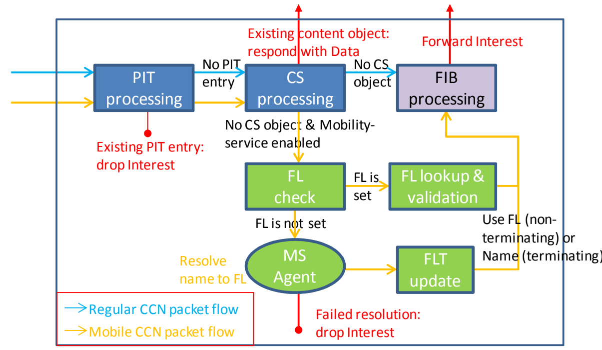

Figure 3: Processing of Interest packets with the proposed architecture.

We can explain the packet processing for the MS-enabled requests in more detail as follows (after no match is found in the PIT and the CS):

- **FL check**: FLPL first checks whether FL is set or not (i.e., whether the Interest carries an FL header).
- **FL lookup and validation**: If FL is set (*i.e.*, request is received by the Producer side ISR or an ICN-R), then FL lookup is performed. At the ISR, if the FL matches the locator name for the receiving ISR, this step outputs the name to use on FIB lookup, which can either be the content name or a new locator (requiring FL swapping).
- **MS-Agent**: Request is forwarded to MS-Agent to resolve the content name to a locator, if FL is not set within the Interest (i.e., request is received by the Consumer side ISR).
- **FLT update**: After name-to-locator mapping is received from the MS-Agent, the resulting mapping is inserted to the FLT, and the locator is forwarded to the FIB at the next step for FIB processing.

The last two steps occur at the ISR, while the first two steps occur at both the ISR and the ICN-R nodes. After the processing finishes, Interest header is updated with the resolved FL and the packet is forwarded accordingly.

2.3 End-to-end Mobility Support

In this section, we explain the procedures used to support content delivery from a mobile Producer, on whose namespace(s) mobility-service is enabled. We illustrate the operations taking place for registration and handover phases in Figure 4, where the Consumer is continually serviced by ISR-3 and the Producer moves from a location serviced by ISR-1 to a location serviced by ISR-2. [7]

[6] Anytime an FLT entry is accessed, a flag associated with the entry is set to indicate that it is refreshed while the associated timeout is reset to the default value.

[7] Note that, proposed solution is considered as a network-driven approach, as the reachability of a mobile host is handled by the network.

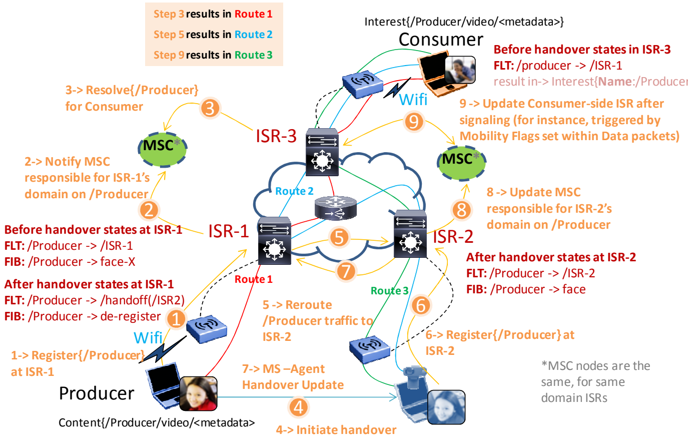

Figure 4: Mobility triggered events taking place before and during handover, including registration and updates.

2.3.1 *Registration Phase*

To register its namespace (referred to as simply /PRODUCER) and trigger the corresponding locator updates, Producer forwards its registration message to the servicing ISR, which is initially ISR-1. After the registration message is received by the ISR-1, it inserts /ISR-1 to its FLT at the hash table bucket corresponding to /PRODUCER. Additionally, ISR-1 also inserts an entry to its FIB to indicate the face corresponding to Producer's location.

Next, ISR-1 updates its MSC with Producer's name-to-locator mapping, which is then conditionally forwarded to Producer's home domain MSC (only if the registration event is triggered by Producer moving into the domain) with a locator of /CURRENT-DOMAIN. Such information is retrieved by ISR-3 from Producer's home domain MSC, when the Consumer makes a first request towards the Producer. If both hosts are located within the same domain, then ISR-3 updates its FLT with an entry of /ISR-1 at the hash table bucket corresponding to /PRODUCER.

2.3.2 *Handover Phase*

In the example scenario, Producer handovers from a point of attachment (PoA) serviced by ISR-1 to a PoA serviced by ISR-2. Specific timings are illustrated in Figure 5, which follows a make-before-break approach. Specifically, Producer is provided with a candidate list of ISRs by ISR-1's MS-Agent before the handover. Here, candidate ISRs are determined based on lower layer information (provided by the Producer) such as the signal-to-noise ratio (SNR) estimates corresponding to in-range PoAs.

–During handover, Producer chooses one of the candidate ISRs as the next ISR (which services the handover'd PoA).

–After the Producer initiates the handover, ISR-1 updates its FLT (through the MS-Agent) to proactively replicate the received Interests towards the candidate ISRs (or to a subset of them) with the updated FL header (while also updating the matching FIB entry to de-register it).

–After the registration message is received by the ISR-2, Registration phase events take place and forwarded Interests can be delivered to Producer over the corresponding interface.

–After the handover, returned Data packets are marked (with the help of the **mobility-update flag** (MU-flag) that can initially be set by ISR-1 to alert Producer of non-optimal path use) to trigger location update at the corresponding ISR node(s). For intra-domain handover, the target for the MU-flag is the ingress ICN router, whereas for inter-domain handover, the target for the MU-flag is the Consumer side ISR. [8]

–NOTE 1: Without additional mechanisms, Interest packets can get lost during handover (during the period referred to as Ω in Figure 5). Various approaches are possible to minimize packet loss for the considered architecture. For instance, we can implement a mobile-PIT (*e.g.*, an extension on PIT) to store such requests and forward them whenever the registration phase is complete. The other option would be to use *optional registration*, during which ISR-1 proactively transfers Producer's namespace before replicating Interests towards the candidate ISRs (bypassing the prefix registration initiated by the Producer).

–NOTE 2: Request redirection during/after handover can trigger u-turned Interests, if the original path overlaps with the path between ISR-1 and ISR-2. In such case, no new state is created at the ICN-R nodes that already has an active entry within PIT.

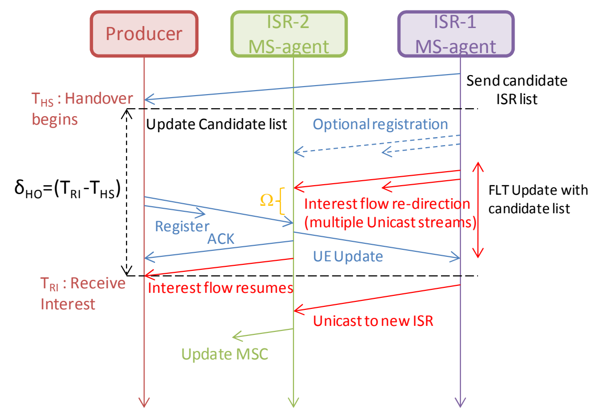

Figure 5: The timings showing events that occur during a mobile handover.

3. PERFORMANCE EVALUATION

To evaluate the performance of our architecture, we used a real-life testing environment similar to one shown in Figure 4 which consists of 3 ISRs, and varying number of ICN-R nodes, with two clients each connected to a different ISR. All the ISR and ICN-R nodes run a multi-threaded TLV-based version of CCNx which is enhanced for mobility support. [9] We used OpenStack to enable mobility service at the ISRs, which host virtualized MS-Agents and Conference Service Agents; and Floodlight controller for the Network, Mobility, and Service Controller functionalities. ISR and ICN-R nodes run on Ubuntu Linux, and are equipped with Intel i7-4770R 3.9GHz CPU and 16GB RAM. Another Ubuntu Linux-based node hosts the controller virtual machines for OpenStack, Floodlight, and JBoss Application Server, all of which are used for the orchestration.

[8] For inter-domain handover, MU-flag can be set before the handover, if Producer is about to move to a different domain, for instance in the case of WiFi to LTE handover.
[9] TLV represents type, length, value.

Producer and Consumer nodes (implemented on laptops) run the same version of CCNx as the ISR and ICN-R nodes. Consumer uses a wired connection to the ISR, whereas the Producer connects with the ISR using a WiFi connection. The Linux *hostapd* tool is used to create the software wireless access point on the ISR nodes.

For the application setup, we considered a real-time video conferencing application between two clients, where the content is delivered in 100ms blocks (as considered for our video conferencing implementation over CCN in [8]). For the given application, Producer sends notifications for each block, and upon receiving the notification, Consumer makes requests for the content chunks.

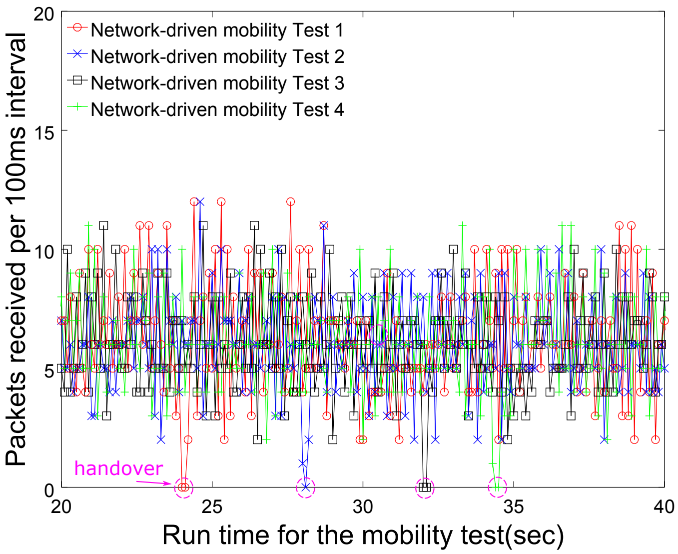

Figure 6: Updated results for network-driven mobility with Producer carrying multiple WiFi adapters (100ms measurement interval).

Based on our initial tests, we observed that the major component for the measured latency is the handover latency caused by lower layer operations, *i.e.*, the time taken to establish WiFi connectivity and the interface to switch from one IP network to another, which is independent of our ICN implementation. As the main focus of our research is to provide seamless mobility guarantees at the ICN layer, we isolated the impact of lower layers by allowing Producer to simultaneously connect to multiple access points and to initiate hand over by switching interfaces (*i.e.*, programming the FIB accordingly).

We illustrate our results with the Producer carrying multiple wireless adapters in Figure 6. The figure shows the packet delivery rate (within 100ms measurement intervals) at the Consumer from four different test runs, with handovers occurring at approximately 24s, 28s, 32s, and 34s. We observed that the direct impact of handover (as connection loss) is limited to at most 2 blocks of data (*i.e.*, latency is upper bounded by 200ms). Upon further examination, we observed that the majority of this latency is caused by the additional switching latency introduced by the CCN, leading to a failure in notification delivery, causing the Consumer to not request packets for the given block.

Due to space limitations, we omit the discussion on scalability and path stretch. However, we observed acceptable performance for both metrics with the proposed solution.

4. CONCLUSION

In this paper, we presented a service-centric mobility support architecture for CCN/NDN-based ICN architectures that is capable of providing seamless mobility support. We provided an in-depth overview of the proposed architecture by presenting a detailed analysis of the architectural components and the procedures required to handle mobility support. We evaluated the performance of the proposed architecture using a video conferencing testbed, and showed promising results towards achieving seamless handover.

5. REFERENCES

[1] 5G-Initiate-Team. NGMN 5G White Paper. Technical report, NGMN Alliance, 2015.

[2] S. S. Adhatarao, J. Chen, M. Arumaithurai, X. Fu, and K. K. Ramakrishnan. Comparison of naming schema in ICN. In *IEEE LANMAN*, 2016.

[3] A. Afanasyev, C. Yi, L. Wang, B. Zhang, and L. Zhang. Map-and-encap for scaling NDN routing. Technical Report NDN-0005, NDN, 2015.

[4] J. Auge, G. Carofiglio, G. Grassi, L. Muscariello, G. Pau, and X. Zeng. Anchor-less producer mobility in ICN. In *ACM ICN*, 2015.

[5] A. Azgin, R. Ravindran, and G. Wang. Mobility study for named data networking in wireless access networks. In *IEEE ICC*, 2014.

[6] A. Azgin, R. Ravindran, and G. Wang. A scalable mobility-centric architecture for named data networking. In *IEEE ICCCN SCENE*, 2014.

[7] D. Farinacci, V. Fuller, D. Meyer, and D. Lewis. The locator/id separation protocol (lisp). RFC 6830, 2013.

[8] A. Jangam, R. Ravindran, A. Chakraborti, X. Wan, and G. Wang. Realtime multi-party video conferencing service over information centric network. In *IEEE ICMEW*, 2015.

[9] R. Ravindran, A. Chakraborti, and A. Azgin. Forwarding-label support in CCN protocol. draft-ravi-ccn-forwarding-label-02, IRTF-ICNRG draft, 2016.

[10] R. Ravindran, S. Lo, X. Zhang, and G. Wang. Supporting seamless mobility in named data networking. In *IEEE ICC*, 2012.

[11] A. Venkataramani, J. Kurose, D. Raychaudhuri, K. Nagaraja, M. Mao, and S. Banerjee. MobilityFirst: A mobility-centric and trustworthy Internet architecture. In *ACM CCR*, Jul 2014.

[12] G. Xylomenos, C. Ververidis, V. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. Katsaros, and G. Polyzos. A survey of information-centric networking research. *IEEE Communications Surveys Tutorials*, 16(2), 2013.

[13] Y. Zhang, H. Zhang, and L. Zhang. Kite: A mobility support scheme for NDN. Technical Report NDN-0020, NDN, 2014.

[14] Z. Zhu, R. Wakikawa, and L. Zhang. A survey of mobility support in the Internet. RFC 6301, 2011.