Content Delivery in the MobilityFirst Future Internet Architecture*

Feixiong Zhang, Kiran Nagaraja, Yanyong Zhang, Dipankar Raychaudhuri
WINLAB, Rutgers University
671 Route 1 South, North Brunswick, NJ 08902
Email: {feixiong, nkiran, yyzhang, ray}@winlab.rutgers.edu

***Abstract*—This paper presents a detailed description of content delivery techniques used in the proposed MobilityFirst (MF) clean-slate future Internet architecture now under development. The MF architecture addresses the requirements of mobile content delivery using the following basic design elements: (1). mapping of human readable names to "flat" globally unique identifiers (GUID's) which are used as the basis for all communication services; (2). a global name resolution service (GNRS) for dynamic binding of GUIDs to network locators; and (3). a hop-by-hop storage-aware transport scheme that exploits in-network storage and copes with characteristics of wireless medium. Technical details for content naming and addressing are given, along with an explanation of methods for publishing, locating and delivering content within this framework. It is shown that the architecture supports seamless migration of content, efficient retrieval and support for optional in-network caching. Ongoing work on a real-time prototype implemented on the ORBIT testbed and the GENI experimental network is also presented.**

I. Introduction

The current Internet has been designed around a point-to-point communication model between two stationary entities based upon their network locators, or IP addresses. Over the course of a few decades, however, Internet usage has evolved to be dominated by content distribution and retrieval. More recently, Internet access from mobile devices has grown dramatically in popularity, and it is anticipated that mobile user generated traffic will exceed that from fixed devices in the next 2-3 years, as reported in a Cisco white paper [1]. These two important trends in Internet usage motivate consideration of efficient content delivery in the future mobile Internet.

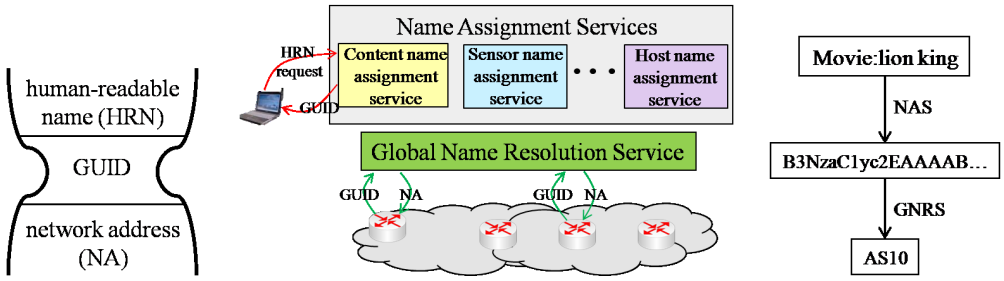
The existing Internet architecture does not provide direct support for either content delivery or mobility, requiring solutions which are overlaid on top of the core IP network protocols. In the traditional Internet, to access content, a host needs to first find out where the content is located. When either the content requestor or the content host becomes mobile, the mapping between the content and its location becomes uncertain. Besides, the end-to-end TCP/IP stack in the existing Internet does not fare well with unreliable, intermittently connected wireless links. Although MobileIP [2] is intended to solve the mobility problem, it has several drawbacks such as triangular routing and limited scalability. Application or service specific solutions, such as content delivery networks (CDN) or peer-to-peer (P2P) systems can be used to ease the burden of supporting content delivery, but they are both application-layer overlay networks and thus less efficient and more costly compared to network-layer solutions.

To address this critical challenge, the MobilityFirst project [3], one of the ongoing NSF Future Internet Architecture (FIA) programs [4], aims to design a clean slate Internet architecture to support large-scale, efficient and robust network services with mobility as the norm. One of the fundamental objectives of MF is to accommodate content-oriented applications and services in mobile scenarios. The following three main features of MF form the basis of our content delivery solution:

1) *Flat Content Name Space*. A flat, self-certifying global unique ID (GUID) space is used to name the content. Each content also has a human readable name. The translation between the human readable name to GUID is managed by content publishers.
2) *GUID to Address Separation*. We decouple content's GUID from its network address at the network level. A distributed global name resolution service (GNRS) is utilized to maintain and resolve the binding between content GUID and address(es). The GNRS is a distributed, peer-to-peer structure between network routers themselves to ensure scalable and fast content discovery.
3) *Reliable Content Transportation*. We employ a hop-by-hop storage-aware transport method for delivering content in mobile and wireless scenarios. Our transport method takes advantage of decreasing cost of router storage and makes it possible to deal with intermittent disconnections and varying wireless channel quality.

In MF, due to name and locator separation, users can request content directly by content name, totally oblivious to its current location or network address. Routers query the GNRS dynamically to get updated content location(s) in a timely fashion, referred to as dynamic binding between name and location, and further choose the nearest content location. Content mobility is thus supported. When delivering content to end users, the MF hop-by-hop storage-aware transport protocol can deal with intermittent and unreliable wireless links. We have built a prototype that realizes this basic content delivery solution with additional support for in-network caching, multi-homing, etc.

*Research supported by NSF Future Internet Architecture (FIA) grant CNS-1040735



(a) naming architecture (b) mapping services (c) an example

Fig. 1: Name-locator separation and mapping services in MobilityFirst

There has been some earlier work on content-aware networking with different proposals on content naming, discovery and delivery. TRAID [5], CCN [6] and subsequent NDN [7] propose to utilize a hierarchical naming structure. Corresponding content routing is based on semantic content names with content-aware routers announcing name prefixes which cover the content that the routers could serve. DONA [8] and LANES [9], however, employ flat and self-certifying content names from cryptographic hash of public key. DONA uses a tree-based content routing approach – routers in DONA form a hierarchical tree, and each router maintains the routing information of all the content published in its descendants.

The rest of this paper is organized as follows. Section II presents the base of our content delivery solution. Section III presents our prototype of the basic design. Finally, Section IV presents the concluding remarks.

II. Content Handling in MobilityFirst

In MobilityFirst, content is considered a first-class network-attached object, a status equal to that of a host. Functions such as addressing content, content discovery and retrieval are therefore explicitly addressed by the network architecture. The architecture also inherently supports content mobility (i.e. relocation) and replication, and are handled in a manner similar to host mobility and multi-homing of hosts, respectively.

Starting with how content is named, the following paragraphs present the basic design that addresses the content life-cycle within MobilityFirst. Towards the end of this paper, we discuss planned extensions to the basic MobilityFirst architecture design with advanced network features that further enhance content functionality.

A. Content Naming and Addressing

MobilityFirst architecture proposes a three-level name-to-locator separation for all network-attached objects including hosts, content, services and even abstract notions such as context. Figure 1 shows the three levels and the corresponding mapping services that enable translation between them.

Human-readable names (HRN) are assigned at the application layer and are published to and maintained by domain-specific services which we call a Name Assignment Service (NAS). For example, content publishers such as Netflix or YouTube can maintain their own NAS that is a HRN-directory of their content, with search functionality made available through public API or subscription based interfaces. One could also envision an integrated NAS service that provides a global directory for all content.

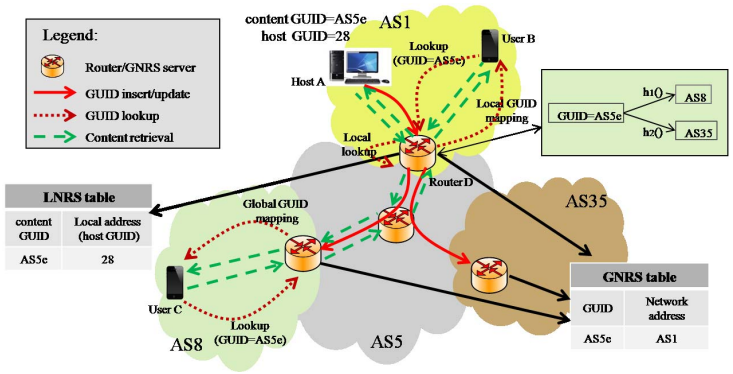
The next level in the separation is a Globally Unique Identifier (GUID) assigned to each network-attached object as its long-lasting network identifier. In addition, a basic design decision in MobilityFirst is to use the GUID as a certification tool. For this purpose, a GUID is a public key, and enables self-certifying communication when passed in packet headers. The mapping from a HRN to GUID is maintained within NAS, and retrieving the GUID for a named network object from the NAS is the first step in communication within MobilityFirst.

The last level in the separation is the locator or network address (NA) that is routable information to reach a network-attached object. The mapping or binding between the long-lasting identifier and the object's current location ($\langle$GUID, NA$\rangle$) is maintained by a globally distributed (logically central) name resolution service (GNRS), a key network support service within MobilityFirst architecture. The syntax of the NA is flexible and defined in cooperation with routing protocols that will use NA to route data packets to the network object. In our base realization of the architecture, NA is a flat network identifier similar to the unique autonomous system (AS) identifier of current Internet.

Therefore, each content object is associated with a HRN, a GUID, and at any time one or more NAs that can be used to route packets to the object. Next we present details of how the $\langle$GUID, NA$\rangle$ mappings for content are published , managed, and lookup during the content life-cycle.

B. Publishing and Locating Content (for Retrieval)

While HRNs for content can be structured (e.g., as URIs), from the perspective of the network, the content name (GUID) is absolutely flat. It is also consistent with how hosts, services, etc., are named within MobilityFirst. This greatly simplifies and unifies the network architecture when supporting tracking, locating and routing requests for content. A publisher looking to make content available over the network first binds it with a GUID. Using the protocol stack, the content is then announced to the network by publishing it's name-to-locator mapping ($\langle$GUID, NA$\rangle$) to the name resolution service - GNRS. The GNRS then serves as the directory for entities looking for any content. Before presenting details of the protocol to publish



(a) content handling overview

Request Packet ← Content GUID, src host GUID, service ID
if *dst in the same AS* **then**
1. resolve content host GUID from local GNRS
2. intradomain routing based on host GUID

else
1. resolve content network address from local GNRS
2. interdomain routing based on network address
3. resolve content host GUID after request reaches the network
4. intradomain routing based on host GUID

end if

(b) content GUID resolution procedure

Fig. 2: Content publish, locating, retrieval and the resolution procedure using two level GNRS

and locate content, we first look at the design of GNRS which is at the core of the content protocol.

GNRS as Content Location Directory The basic operations supported by GNRS are *insert*, *update* and *lookup*, and they correspond to a fresh insert of ⟨GUID, NA⟩ mapping, an update to an existing mapping, and the lookup of a content's location, respectively. To support response latencies lower than 100ms for these operations (we postulate this is required to enable real-time apps for mobile hosts without loss in quality), the GNRS is designed as an in-network 1-hop DHT with server instances co-located with the routing elements. Each server in this distributed service advertises responsibility to a portion of an orthogonal name-space, which is used to designate (using consistent hashing) a host for mappings of a particular GUID. The service may also employ multiple hash functions to arrive at more than one host for a mapping - enables lower latencies by retrieving mapping from nearest host. The tradeoff, of course, is the cost of managing consistency across these replicas. In one manifestation of this GNRS scheme[10], BGP prefix announcements from individual ASes are used to determine the identity of the hosting AS. Detailed evaluations of this scheme with real data from Internet topologies have shown that the latency target is achievable with only a small number of replicas (five) to host each mapping.

Two-level Structuring of GNRS for Content In routing to a network attached object, the NA part of a GNRS mapping may contain information for what is traditionally thought as inter-domain routing, as well as information required to route locally within a domain. If it contains local routing information then the downside is that mobility within a domain also results in updates to GNRS. For this reason, local routing in Mobility-First is carried out on the GUID of the network attached object, requiring only a network identifier in NA at the GNRS. This hybrid GUID and network address routing (HGN) approach provides a flexible and robust routing scheme for mobile Internet. For content, however, it's GUID is excluded from local routing state, requiring the local address information at GNRS. Instead, we avoid accesses to GNRS for changes to content's local address(LA) by introducing a Local Name Resolution Service(LNRS) within each domain. This design decision has the following implications to name resolution:

- Location updates for content are first sent to LNRS to store the ⟨GUID, LA⟩ mapping. The LNRS then forwards the request to a GNRS server to store the ⟨GUID, NA⟩ mapping where NA is the network name.
- When a content moves within the local network, only the local address mapping needs to be updated (at LNRS), while global mapping is unchanged. This reduces the request load at GNRS plane significantly.
- A lookup request for a content GUID is first sent to a LNRS, which may respond directly if the GUID is present locally. The request need be forwarded to a GNRS server only when the content is not local.

Caching Name Resolution Mappings To allow caching of name-to-locator mappings at points in the name resolution structure, an expiry time is assigned to each mapping. This expiry duration is determined by the content publisher and is taken as an informed estimate. The publisher is required to refresh the mapping as required to prevent the mapping from expiring. The expiry field is used by intermediate resolution servers to make caching decisions and also when responding to requesters. Note that unrealistic settings of the expiry time result in expected downsides of either increased refresh updates when overly conservative, or stale mappings when overly optimistic. Publishing entities can also prevent caching of the mappings through directives at publish time.

Figure 2a shows the role of two-level structured GNRS where host A publishes a content while hosts B, C request the content from AS1 and AS8 respectively. Note that router D does a local GUID lookup on receiving request from C.

Publish Protocol When a content is published, a GNRS *insert message*: ⟨GUID, Addr⟩ is sent to an LNRS server established at the time of network attachment. If no such server is configured, the message is sent to default gateway which then forwards the message to LNRS. The LNRS server on receiving the insert message first initiates an update at the local service plane with ⟨GUID, LA⟩. It then applies a consistent hash function on the GUID to determine the GNRS server that will host the mapping ⟨GUID, NA⟩. An insert message is then forwarded towards the network or AS containing the GNRS server to host the mapping. An ACK message confirming the insert operation is returned to the content host.

Locating and Requesting Content In our basic design, the location of a content is established by directly interacting with name resolution service. This is by initiating a GNRS *lookup message*: ⟨GUID, options⟩ to LNRS. The LNRS server first checks with the local service plane whether the request is for a content located within the local network. If so the request need not be forwarded to the global plane, and a response is returned from the LNRS server hosting the mapping for the GUID. If content is not local, a consistent hash determines the GNRS server that hosts the mapping, and the lookup message is forwarded. Address received in the response to the lookup request can then be used to address a content retrieval request. Figure 2b presents this resolution procedure.

A second approach is more direct, and exploits the network's ability to dynamically resolve location for a given GUID. The network API designed for the MobilityFirst architecture, implements a *get(GUID, buffer, options)* interface, where GUID refers to the content's id. The resulting request packet is handled by a router by first resolving the location of the content (through GNRS), and then forwarding the packet forwarded directly to the content's latest location.

Content Mobility and Stale Mappings When a content or it's host moves, the host initiates a *update message*: ⟨GUID, Addr, options⟩ to LNRS. If the update is only to the local address, the update is handled at LNRS plane alone. Else the LNRS forwards an update message to the GNRS server determined by the hash, to effect changes to the global mapping.

When the movement is across networks, the mappings at LNRS of previous network should be invalidated. The expiry time of the mapping is one mechanism to address this issues. A carefully chosen value can effectively minimize the staleness window. However, this is not always possible. In a managed handoff solution, the content host sends an update message prior to leaving the network to either invalidate the mapping, or to append a projected network path. A third approach which is expensive to support is for GNRS to trigger invalidations at LNRS at previous network of the moved host.

Finally, when packets are routed using stale address information, failures result in two actions: (1) automatic invalidation of stale mapping; and (2) fresh lookup requests to the GNRS plane for up-to-date address information.

C. Robust Data Delivery

Once a content request reaches the content host, the hosting app can initiate content transfer to the requester using the *send(GUID, message, service options)* interface supported by the network API. The GUID here refers to the requester's network id, and the message is either the entire content or a segment with more to follow in subsequent iterations.

The MobilityFirst protocol stack implements robust and efficient message delivery by combining hop-by-hop delivery of large data blocks [11] with storage-aware routing within the network. This contrasts with TCP/IP protocol stack in the following important ways:

- data blocks are reliably transferred hop-by-hop, considerably reducing end-to-end signaling
- data blocks routed by MobilityFirst routers are chunks of upto 100s of MB in size, as opposed to MTU-sized packets of present Internet protocols
- temporary store for data blocks in routers to overcome intermittent link fluctuations and host disconnections

We believe that these key differences enable our delivery architecture to perform better than the TCP/IP protocol stack, especially for wireless access networks with unstable link quality and even frequent disconnections. While our protocol stack necessitates only a thin transport layer, additional end-to-end robustness can be employed over this delivery platform through suitable transport implementations added to the protocol stack. Below, we present two key components of our content delivery architecture, with details on how data blocks are transported through the network.

Fig. 3: Chunk 'packet' with key fields. While GUID is the authoritative routing information, the NA (if bound) provides fast forwarding path.

Reliable Hop-by-Hop Chunk Transfer The transport layer is responsible for taking a message and segmenting it into large blocks called chunks. A chunk represents an autonomous data unit that is routed through the network (hence also called a PDU), and therefore contains the header with authoritative routing information - the destination GUID, as seen in Figure 3. A chunk can be as large as a few hundred MBytes, but the size can also be negotiable with the next-hop or even the final recipient of the message, to accommodate resource differences.

A chunk can be routed entirely on GUID as is the case when destination is in local domain. Alternately, the NA of the destination can be resolved by the routing layer by doing a GNRS lookup. Once NA is bound, a chunk can be routed along a fast path until further lookups are necessary due to failures, for example. A chunk ready for transfer is then handed to the link layer, where it is fragmented and transmitted as MTU-size frames in a hop-by-hop fashion. The link layer at the next hop aggregates the entire chunk before passing it to the routing layer to be routed on either the GUID or NA. A chunk that fails to transfer to the next hop is handed back to routing layer for rerouting or temporary storage.

Storage-Aware Routing In MobilityFirst, we've proposed a generalized storage-aware routing protocol (GSTAR)[12] that can seamlessly adapt among and work across varied spectrum of wired, wireless and even DTN-type networks where partitioning and disconnections are quite common. It exploits storage available at each router to overcome disconnections and intermittent link quality variation, esp. in mobile wireless or congestion scenarios. GSTAR uses short term ETT (most recent few estimates) and long term ETT(statistic over a longer period) to make a relative estimation of current link quality where ETT is derived from periodic probe messages sent to neighbors. Aggregated ETT values are also propagated multiple hops through periodic link state announcements(LSA). If

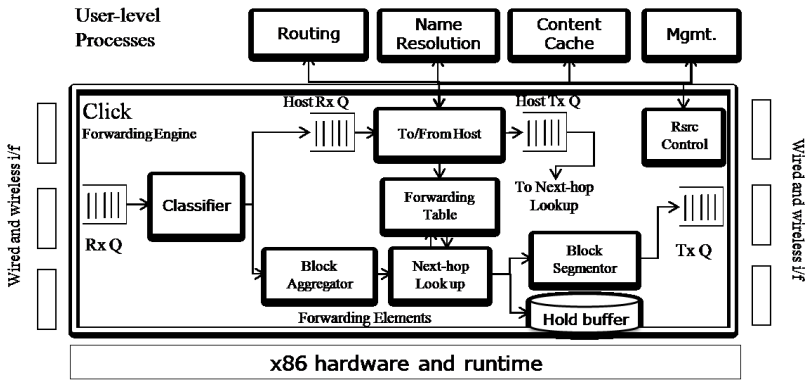


Fig. 4: A component view of Click-based router prototype. Content caching module is being implemented as a user-level process outside of Click and will receive content requests and data packets intercepted by the forwarding engine.

link/path quality is better or normal,the chunks are forwarded along their way. If not, the chunk is held in a local store until link quality improves. Chunks are also held in the store during host disconnections as is the case when hosts move across networks or change access points.

GSTAR also interacts with GNRS to resolve NA(s) for chunks with only GUID information in their headers. A GNRS lookup is also required when hosts move out of the network and connect to a different network, resulting in local delivery failures. Low latency requirements for GNRS make dynamic binding between GUID and NA possible, even multiple times in transit. Mobility is therefore a core concern and supported as a normal event within our content delivery architecture.

III. Prototype

We have implemented proof-of-concept prototypes of the network and host components including a Click-based router[13], the distributed name resolution service, and the host-side protocol stack and network API. The router implements reliable hop-by-hop data transport, storage aware routing, and an early version of a content caching module to support in-network content caching.

As seen in Figure 4, Click elements implement the data forwarding path, while routing control, management and services such as name resolution and content caching are implemented as user-level processes. Router processes send and receive messages with peers via the Click forwarding engine. The name resolution is structured into two service planes (LNRS and GNRS) as described in Section II. The host protocol stack that implements the Hop data transport and a GUID services layer has been prototyped for the Linux and Android platforms and interacts with MobilityFirst network routers over Ethernet, WiFi, and WiMAX links. A network API implemented as a user-library provides GUID-based *send* and *receive* messaging interfaces. It also implements *publish* and *get* interfaces to publish, locate and retrieve content.

These prototypes are currently under evaluation within the ORBIT grid testbed[14] which has a total of about 400 nodes. Our longer-term objective is to have the prototype network run over the nation-wide GENI testbed[15] for at-scale evaluation and validation of protocol architecture.

IV. Conclusions and Future Work

In this paper, we have presented our mobility-centric content-aware networking scheme to meet the challenge of mobile content delivery. We propose to use a flat self-certifying GUID to name content and present a clean separation of name from location(s). To support fast name-to-location resolution we propose a distributed in-network name resolution service (GNRS) that employs a 1-hop hashing technique. We then design a storage-aware routing architecture that reliably transfers data blocks hop-by-hop, and by consulting GNRS can dynamically bind GUID to network location to enable efficient and robust content delivery to mobile hosts. We also present details of proof-of-concept prototypes and wide-area evaluations of our baseline architecture. Several extensions to the baseline architecture including support for in-network caching are under consideration.

References

[1] Cisco: Global Mobile Data Traffic Forecast Update, 2009-2014.
[2] C. Perkins, "IP mobility support for ipv4, revised," *IETF Internet Standard, RFC 5944*, Nov 2010.
[3] MobilityFirst project, http://mobilityfirst.winlab.rutgers.edu/.
[4] NSF FIA project, http://www.nets-fia.net.
[5] M. Gritter and D. R. Cheriton, "An architecture for content routing support in the internet," in *Proceedings of the 3rd conference on USENIX Symposium on Internet Technologies and Systems*, 2001.
[6] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *CoNEXT '09*, 2009.
[7] Named data networking (NDN) project, www.named-data.org.
[8] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," in *SIGCOMM '07*, 2007.
[9] K. Visala, D. Lagutin, and S. Tarkoma, "LANES: an inter-domain data-oriented routing architecture," in *ReArch '09*. ACM, 2009.
[10] T. Vu, A. Baid, Y. Zhang, T. D. Nguyen, J. Fukuyama, R. P. Martin, and D. Raychaudhuri, "Dmap: A shared hosting scheme for dynamic identifier to locator mappings in the global internet," in *ICDCS 2012*.
[11] M. Li, D. Agrawal, D. Ganesan, and A. Venkataramani, "Block-switched networks: A new paradigm for wireless transport," in *Proc. of NSDI*, 2009.
[12] S. C. Nelson, G. Bhanage, and D. Raychaudhuri, "GSTAR: generalized storage-aware routing for mobilityfirst in the future mobile internet," in *MobiArch '11*. New York, NY, USA: ACM, 2011.
[13] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The Click Modular Router," *ACM Trans. on Computer Systems*, aug 2000.
[14] ORBIT testbed, http://www.orbit-lab.org/.
[15] Global Environment for Networking Innovations (GENI), www.geni.net.