MobilityFirst: A Robust and Trustworthy Mobility-Centric Architecture for the Future Internet

Dipankar Raychaudhuri, Kiran Nagaraja
WINLAB, Rutgers University
Technology Centre of NJ, 671 Route 1
North Brunswick, NJ 08902
ray, nkiran@winlab.rutgers.edu

Arun Venkataramani
Department of Computer Science
University of Massachusetts
Amherst, MA01003
arun@cs.umass.edu

***Abstract*[1] —This paper presents an overview of the MobilityFirst network architecture, currently under development as part of the US National Science Foundation's Future Internet Architecture (FIA) program. The proposed architecture is intended to directly address the challenges of wireless access and mobility at scale, while also providing new services needed for emerging mobile Internet application scenarios. After briefly outlining the original design goals of the project, we provide a discussion of the main architectural concepts behind the network design, identifying key features such as separation of names from addresses, public-key based globally unique identifiers (GUIDs) for named objects, global name resolution service (GNRS) for dynamic binding of names to addresses, storage-aware routing and late binding, content- and context-aware services, optional in-network compute layer, and so on. This is followed by a brief description of the MobilityFirst protocol stack as a whole, along with an explanation of how the protocol works at end-user devices and inside network routers. Examples of specific advanced services supported by the protocol stack, including multi-homing, mobility with disconnection, and content retrieval/caching are given for illustration. Further design details of two key protocol components, the GNRS name resolution service and the GSTAR routing protocol, are also described along with sample results from evaluation. In conclusion, a brief description of an ongoing multi-site experimental proof-of-concept deployment of the MobilityFirst protocol stack on the GENI testbed is provided.**

Keywords- Future Internet architecture, mobile networks, name resolution, storage-aware routing, GENI prototyping.

1. INTRODUCTION

The *MobilityFirst* architecture described in this paper is founded on the premise that the Internet is approaching an historic inflection point, with mobile platforms and applications poised to replace the fixed-host/server model that has dominated the Internet since its inception. With over 6 billion cellular mobile devices in worldwide use today, it is anticipated that by 2015, mobile data devices will significantly outnumber fixed hosts on the Internet. This predictable, yet fundamental, shift presents a unique opportunity to design and develop a next generation Internet architecture in which mobile devices, mobile applications, and the consequent changes in service, trustworthiness, and management are primary drivers of a new architecture. The MobilityFirst architecture, while inspired by this historic shift, is nonetheless informed by research and experience with today's Internet architecture, and offers significant benefits to wired networks and users as well.

Why should mobility come "first" as we contemplate a clean-slate redesign of the Internet architecture? The simple answer to this is the fact that the number of mobile devices and their traffic are growing at a remarkable exponential rate and are poised to surpass all other Internet traffic in just a few years. To quote from a recent Cisco white paper [1], "*Traffic from wireless devices will exceed traffic from wired devices by 2014. In 2016, wired devices will account for 39 percent of IP traffic, while Wi-Fi and mobile devices will account for 61 percent of IP traffic.*" These numbers confirm the emergence of what is popularly known as the "mobile Internet". This will inevitably drive a gradual convergence of cellular networks with the Internet both in terms of business models and technical standards. The challenge for network architects is to effectively merge two very different network designs into a unified network architecture that efficiently supports billions of portable devices running new classes of mobility applications in a trustworthy manner. Looking ahead another 5 years to ~2020, the mobile Internet will not be limited to cellular, but will also include a variety of wireless sensor, machine-to-machine (M2M), smart grid and vehicular (V2V) scenarios

[1] Research supported by NSF FIA (Future Internet Architecture) grant CNS- #1040735

associated with integration of physical world awareness and control into Internet applications [2][21]. Such "pervasive" or "ubiquitous" wireless scenarios pose additional architectural challenges, for example dealing with frequent disconnections, energy constraints or providing strong security for real-time control applications.

Our vision for a clean-slate redesign of the Internet is not to simply "fix" the current Internet Protocol (IP) with a better and more secure design, but to use this opportunity to fundamentally re-evaluate the purpose, functionality and trustworthiness of the network in the all-important context of mobility everywhere. Although the current Internet protocol suite has been remarkably successful for several decades, it was designed for end-user services and technology assumptions that are not well-matched to mobile devices. For example, IP address assignments are designed to be static identifiers of network location and TCP assumes a contemporaneous end-to-end path - assumptions often violated in mobile scenarios. Today, the cellular network has become the first point of attachment for many mobile devices, however, this access network is based on an addressing and transport architecture derived from circuit-switched technologies, requiring the use of service gateways for bridging cellular services to the Internet.

Although cellular-IP gateways may be a workable solution in the short run, there are significant scalability, performance, management, and security problems when bridging two architecturally different networks. Furthermore, new services (such as disconnection tolerant delivery, content caching or context-aware multicast) that are not well matched to today's architecture generally require custom overlays and/or protocol gateways. Widespread use of overlays can fragment the Internet into multiple application-specific domains, reducing network-effect benefits for both developers and end-users alike. Therefore, we envision a future Internet architecture that supports mobile devices as "first-class" objects without the need for gateways or overlays, thereby enabling a variety of new services and applications efficiently, securely, and at scale.

In Section 2 that follows, we first identify a set of high-level design goals and then describe a specific network architecture needed to achieve these requirements. An overview of the MobilityFirst architecture is given along with a discussion of the key features of the proposed protocol stack. This is followed by an explanation of how the protocol works at end-points and network routers, with examples of specific services such as mobility, multi-homing and content retrieval. Next we provide a brief discussion in Section 3 of some of the key protocol components in the proposed architecture, most notably the global name resolution service (GNRS) and generalized storage-aware routing (GSTAR) routing protocol. Finally, Section 4 provides a status update on ongoing proof-of-concept prototyping activities using the GENI (Global Environment for Network Innovation) testbed as the platform [16].

2. MOBILITYFIRST ARCHITECTURE

2.1 Design goals: The MobilityFirst architecture is centered on two fundamental goals: *mobility* and *trustworthiness*. The mechanisms used to realize these high-level goals in MobilityFirst are also mutually reinforcing, i.e., some of the mechanisms used to improve mobility also enhance trustworthiness. To appreciate this point, we begin with a list of the high-level design goals that drive the design of MobilityFirst (and that are not adequately met by the current Internet):

1) *Seamless host and network mobility*: The architecture should seamlessly support mobile devices as well as networks at scale. Mobility and the presence of wireless links should be considered the norm. In contrast, the current Internet is primarily designed with tethered hosts in mind, e.g., an IP address is used to identify a host/interface as well as its network location. This makes it cumbersome to support mobility (when a host's network location keeps changing) as well as multi-homing (when a host is simultaneously attached to multiple network locations).

2) *No single root of trust*: The architecture should not require a single global root of trust. In contrast, the current Internet has a single authority (ICANN) that must be trusted in order to reliably translate names to IP addresses [3].

3) *Intentional data receipt*: Receivers should have the ability to control incoming traffic and, in particular, be able to refuse unwanted traffic [4]. In contrast, the current Internet largely treats receivers as passive nodes that have little control over the traffic sent to them.

4) *Proportional robustness*: A small number of compromised nodes must not be able to inflict a disproportionately large impact on the performance or availability of the rest of the nodes.

5) *Content addressability*: The network should facilitate content retrieval by allowing addressing of content independent of its hosted location.

6) *Evolvability*: The architecture should allow for rapid deployment of new network services.

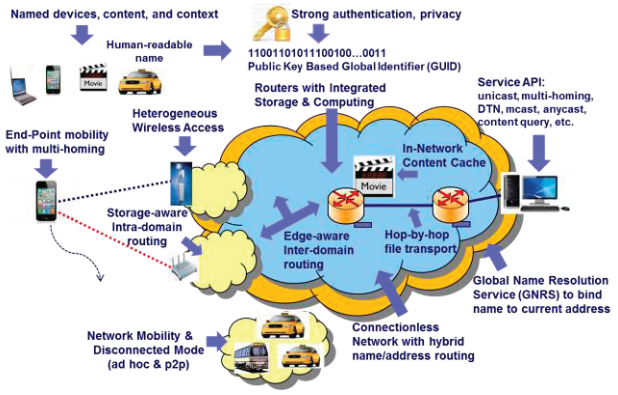

Fig. 1: Major design features of MobilityFirst architecture

2.2 Architecture Concepts: The architectural research conducted in this project started with the abstract requirements outlined above and then considered high-level protocol design approaches towards realizing these goals. The MobilityFirst architecture (shown in Figure 1) that emerged through gradual consensus, includes a system concept and a set of protocol elements, and is centered on a new name-based service layer which serves as the "narrow-waist" of the protocol stack. The name-based service layer uses the concept of "flat" globally unique identifiers (GUIDs) for network attached objects, a single abstraction that covers a broad range of communicating objects from a simple device such as a smartphone, a person, a group of devices/people, content or even context. GUIDs are basically public keys assigned by a name certification service to the networks objects and are the long-lasting network-level identifiers for these objects. The name-based service layer uses the GUIDs to both enable mobility-centric services and also to be the crux of the architecture's security and trustworthy properties.

Network services invoked on messages are defined first and foremost by the source and destination GUIDs, and next, quite distinctly, by a service identifier (SID) that specifies the delivery mode such as unicast (default), multicast, anycast, multi-homed, content retrieval or context-based message delivery. For routing, a hybrid name/address based scheme is used for scalability, employing a fast global name resolution service (GNRS) to dynamically bind the destination GUID to a current set of network addresses (NAs). The GNRS thus forms a central feature of the mobility-centric architecture, enabling on-the-fly binding of names to routable addresses as needed for dynamic mobility, disconnection or cloud migration scenarios. Actual delivery of data through the network is based on hop-by-hop transfer of large data blocks while leveraging in-network storage to deal with link quality variations and disconnections due to mobility. The corresponding intra- and inter-domain routing protocols used in the network have new features such as edge-network awareness and *late-binding* (i.e., binding or re-binding of GUID to NA(s) other than at the source) capabilities needed to achieve the design goals. The overall philosophy of the design is thus back to the basics of packet switching with hop-by-hop routing of data blocks, which are entirely self-contained with authoritative routing information, with a minimum of in-network state.

Some of the major design features of the architecture are discussed further below:

Separation of Names and Addresses: MobilityFirst cleanly separates human-readable names, the corresponding globally unique identifiers, and the dynamic network address locators. In contrast to the current Internet, the human-readable name can be managed and assigned to a unique GUID by multiple name certification services (NCSs) without a global root of trust. No coordination is required between NCS providers because the GUID space is very large with negligible probability of collisions. GUIDs assigned to network objects are mapped to a set of network addresses (NAs) or locators corresponding to the current points of attachment.

Security based on Verifiable Globally Unique Identifier: The GUID assigned by an NCS is derived from a public key thereby enabling authentication and security services in the network. Deriving the GUID as a cryptographic hash of a public key also enables them to be self-certifying, i.e., authenticating a node does not require an external authority [5, 6].

Name-based Network Service API: The service API in MobilityFirst is based on the names of source and destination network objects, rather than on the network addresses/interfaces. This allows us to build abstract services involving multi-homed devices, groups of objects, named content, etc. Because a single network object may consist of multiple devices or have multiple interfaces, the service abstraction

is inherently multicast in nature and is thus well matched to the wireless environment.

Hybrid Name/Address Based Routing: The proposed architecture uses hybrid name/address based routing to achieve scalability. The total name space of attached network objects is expected to be on the order of ~10-100 billion, while the number of unique routable networks is expected to be in the order of millions, thus making it expedient to map GUIDs to NAs and then route using NAs where available. This approach requires the existence of a global service (which we call the GNRS) that dynamically maps GUIDs to NAs. Clearly, scale and speed of the GNRS are critical design requirements for the proposed approach to be viable for mobility services.

Mobile End-Points with Multi-Homing: Our future Internet design assumes the existence of billions of mobile end-points each of which traverses as many as hundreds of wireless access networks in a day. In addition, mobile end-points will typically be multi-homed devices with multiple network interfaces enabling simultaneous access to multiple wireless networks such as WiFi and cellular. Name based message delivery, where a single device GUID can be used to address packets to any of a multi-homed device's current network attachments, makes it possible to offer seamless mobility and multi-homing services without the problems associated with today's IP networking.

Network Mobility, Ad-Hoc and Disconnected Modes: The MobilityFirst protocol stack is also being designed to support network mobility, i.e. migration of entire networks and not just end-points. In addition, the network should support ad hoc infrastructure-less communication between mobile devices in proximity (for example vehicle-to-vehicle) without the need for a connection to the Internet. Thus the name resolution and routing protocols are designed to deal with periods of disconnection in a robust manner.

Storage-Aware Intra-domain and Edge-Aware Inter-Domain Routing: MobilityFirst intra-domain routing protocols are designed to support in-network storage when necessary to overcome link quality fluctuations and disconnection. In addition, the global inter-domain routing protocol needs to have some degree of edge awareness because of the need to deliver data efficiently to/from multiple edge networks with very different properties, e.g. slow cellular vs. fast wired.

Hop-by-Hop Transport: The MobilityFirst protocol uses hop-by-hop (or segment-by-segment) transfer of large files between routers, with the entire file received and stored at each node before sending on to the next hop [7]. This approach makes it possible to implement storage and late binding functions at routers, while also providing important performance benefits (over conventional flows with end-to-end transport) in dynamic wireless/mobile environments.

Optional in-network computing services: Handling of messages/files at routers makes it possible to introduce enhanced services via an optional computing layer at the routers. This computing layer can be invoked for certain GUIDs and SIDs, enabling functions such as content caching, location-aware routing, or context-aware message delivery. This feature also offers a path for evolution of protocol functionality over time.

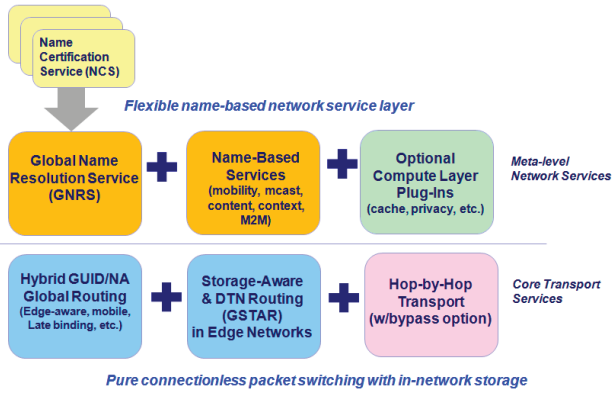

Fig. 2: Basic building blocks in MobilityFirst

The architecture outlined above can be realized with the following basic protocol building blocks, summarized in Figure 2. In addition to name certification services (NCS) shown at the top level, the protocol design involves two distinct layers, the meta-level network services layer responsible for realization of abstract name-based services and a core transport services layer responsible for routing and forwarding. Meta-level network services are implemented as three basic modules – the global name resolution service (GNRS) which is a distributed service across the whole network, the name-based service layer as part of protocol stack on all end-points and routers, and optional compute layer plug-ins at participating routers. Core transport services are also implemented as three distinct modules – the hybrid GUID/NA based global routing protocol, storage-aware/DTN intra-domain routing and hop-by-hop transport.

2.3 MobilityFirst Protocol Overview: Based on the considerations outlined in Sections 2.1 and 2.2, we have developed an initial specification (v1.0) for the high-level protocol architecture of the network. Although the design of the MobilityFirst architecture is still evolving and details are being added at each level, there is a general understanding of the packet structure, major header elements, primary protocol mechanisms (such as name resolution and routing), service types and major security mechanisms. The reference MobilityFirst protocol stack as currently defined is shown in Figure 3. As mentioned earlier, the protocol stack is centered on the GUID service layer which provides abstractions for name-based services. The GUID service layer in the data plane is supported by the GNRS in the control plane, while the routing functions in the data plane are supported by intra- and inter-domain control protocols for routing. Note also the optional compute services layer available above the GUID service layer. Next above are multiple end-to-end transport protocol options which provide socket API's for services such as multicast message delivery, delayed delivery, and content retrieval. Applications are supported in the control plane by the name certification services which provide GUIDs corresponding to the human-readable names of named objects.

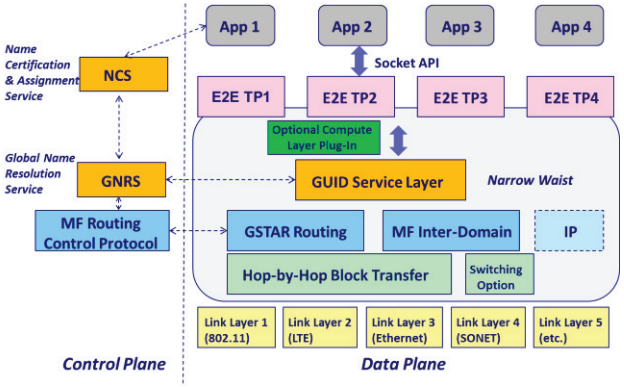

Fig. 3: MobilityFirst protocol stack

In order to understand how the protocol works in further detail, first consider how a name is converted into a GUID by the name certification service (NCS). As shown in Figure 4, a number of specialized NCS providers may cater to name assignment and trust management in different domains such as devices/hosts, M2M, content or context. There may also be a network naming and trust services from which constituent networks obtain their GUIDs and build trust relationships with other peer networks.

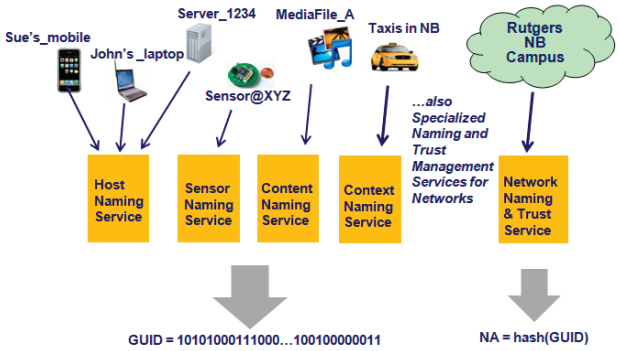

Fig. 4: Multiple NCS providers in MobilityFirst

Next, consider how a message is sent between two end points. As shown in Figure 5, a host wishing to send a message to all of "John's devices" will obtain the corresponding GUID from either the NCS or the end-user and then invoke the service API using a command such as *send (GUID, options, data)*, where options can include service features such as anycast, multicast, timed delivery and so on. The host stack then prepares a MobilityFirst packet with GUID and SID in the header as shown. The GUID is then resolved through a GNRS lookup (either at the sending host or at the edge router) to the set of network addresses (NAs) corresponding to the current points of attachment of this abstract object, in this case NA99 and NA32. The packet header actually sent out by the host (or edge router) then consists of a destination GUID, SID and list of NAs.

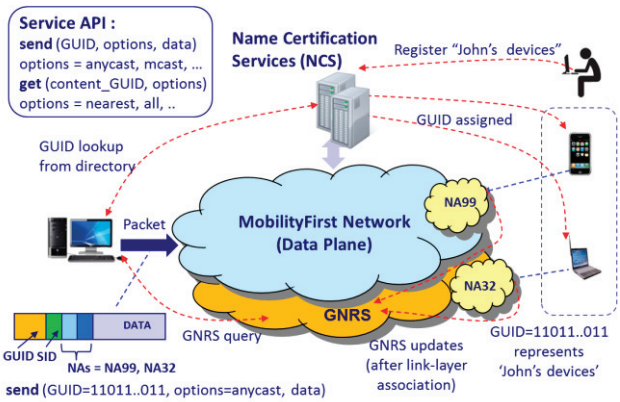

Fig. 5: Steps in message delivery between end-points

Routers in the network will use the NAs (which can be thought of as a "fast path") to make forwarding decisions, with multicast and copy functions added in where appropriate to reach both NA99 and NA32. If a delivery failure occurs due to disconnection or mobility, the packet is stored inside the network and

the GNRS is periodically queried for a rebinding of the GUID with NAs. Depending on requested delivery service (e.g., delay tolerant) and local policy, the packet is either delivered to the new destination within a certain amount of time, or discarded due to time-out.

Consider next the actions at a MobilityFirst router, as shown in Figure 6. Each router in the network has access to two kinds of routing tables – one that maps the GUID to NAs (implemented as a virtual DHT table as discussed later), and other that maps the destination NA to a next hop or port number for forwarding. From the figure, it is observed that packets entering the network always have the destination (and source) GUID attached to the protocol data unit (PDU). There is also a service identifier (SID) in the packet header which indicates the type of service required by the PDU including options such as unicast, multicast, anycast, context delivery, content query, etc. As explained earlier, there may also be an optional list of NAs appended to the GUID and SID in the packet header.

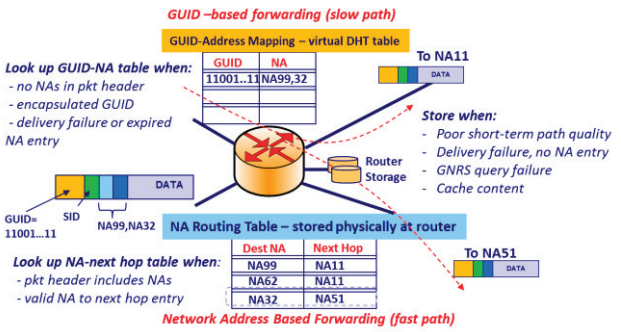

Fig. 6: Hybrid GUID/NA packet processing at routers

When a list of resolved NAs corresponding to the GUID is available, the router only needs to look up the NA routing table as in a conventional IP router – this is referred to as "fast path" forwarding. Any router along the path has the option of resolving the GUID again by querying the GNRS – this is the so-called "slow path" that allows for rebinding to a new set of NAs that may have resulted from mobility or temporary disconnection. The GUID routing option makes it possible to implement "late binding" algorithms where the decision on which NAs for routing can be identified or modified while the PDU is in transit. Of course, there is a higher cost for GUID lookups at routers due to latency and GNRS protocol overhead, but this is incurred only once for large protocol data units which may be ~10M-1GB in size. Note that both GUIDs and NAs in the architecture are flat, i.e. no hierarchical structure is assumed.

MobilityFirst projects and leverages the ready availability of sizeable in-network storage in future routers. Each router along the path has the option of temporarily storing PDUs at routers instead of forwarding towards the destination when poor link quality or disconnection is detected by the path quality metric. PDUs that are placed in temporary storage are scheduled for periodic checks of path quality and forwarded when appropriate. If the destination is completely disconnected, the router will also periodically initiate GNRS queries to determine the new point of attachment if any. Also, a reliable hop-by-hop transport protocol is used to deliver messages between routers in contrast to the end-to-end approach used in TCP/IP.

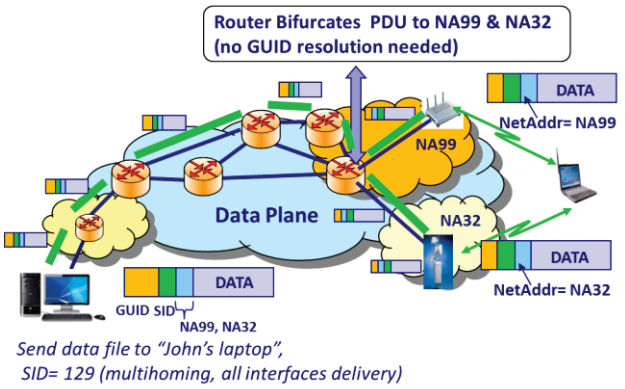

Fig. 7: Handling of dual-homing within MobilityFirst. Data can be delivered to both interfaces of end point by specifying a single device GUID.

Beyond robust message delivery to mobile endpoints, MobilityFirst embeds a flexible and extensible in-network service architecture, with particular emphasis on multicasting and anycasting modes as integral capabilities of the routing protocol. These service features have been provided in response to the needs of mobility applications that often care more about the context (e.g. device location or function) than its network address. The GUID mechanism outlined above allows for context and content addressability with multicasting or anycasting to the set of network addresses associated with a GUID (such as taxis in New Brunswick or Alice's laptop). A particularly interesting use case that is difficult to handle with conventional IP is that of "dual-homing" where a user's laptop may have two wireless interfaces (such as WiFi and 3G) on separate access networks, and the service objective is to deliver to at least one of these interfaces based on a suitable cost

metric. An example of how the protocol works for such a dual-homing scenario is given in Figure 7. In this example, the GUID for "Alice's laptop" is resolved to two distinct network addresses corresponding to the 3G and WiFi networks that it is currently connected to. The PDU carries both these network addresses and the network routing protocol implements a "longest common path" forwarding algorithm where the original PDU is forwarded without duplication as far as possible before bifurcating (copying) to both destinations.

Another service example is given in Figure 8, which shows the case with mobility and complete disconnection of the end point. In this example, the initial binding of the GUID is to NA99, but a delivery failure occurs at the edge network due to device mobility. The protocol data unit is then stored at the edge router and the GNRS is periodically queried for the new NA, which later turns out to be NA75. The packet is then updated to include the new NA and then forwarded towards the revised destination network.

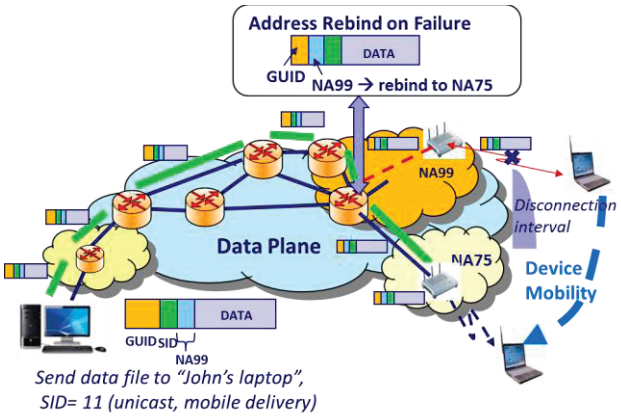

Fig. 8: Example of mobile service under temporary disconnection. Delivery failure at NA99 due to device mobility triggers an address rebind to NA75 – device's new network.

As a final example, Figure 9 shows an enhanced content caching service using the optional compute layer. In this case, a mobile device wishing to retrieve content simply makes a query such as *get (Content_GUID, options=cache_service + nearest)*. Through the GNRS, the content's GUID is resolved to a set of NAs (NA 99, 31, 22 and 43) where the content is currently cached. The access router then looks up the closest cache location from the routing table and forwards the query to the applicable network (NA99). The caching router which offers the enhanced service processes this query at the computing layer and then sends back the requested content to the mobile device as shown. Other services such as location-based message forwarding can also be implemented in a similar manner.

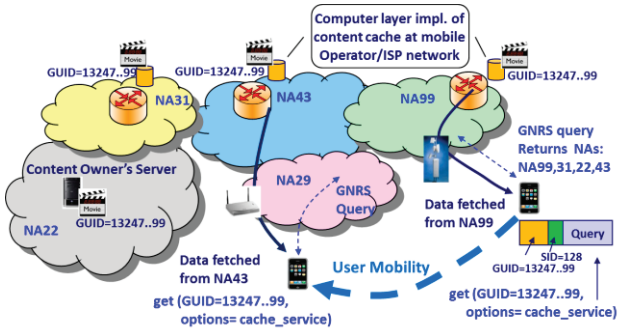

Fig. 9: Example of an enhanced service (content caching) using the compute layer. The network determines the nearest location of content's cached copy for retrieval.

3. Name Resolution & Routing

In this section, we briefly review two of the basic protocol components needed to realize the MobilityFirst protocol stack outlined in Section 2, namely the global name resolution service (GNRS) and the generalized storage-aware routing (GSTAR).

3.1 GNRS: As discussed earlier, the GNRS is a global service which provides the dynamic binding between the name of a network-attached object (i.e., it's GUID) and its current network addresses or locators. There are two challenges in this design – the first is the large scale with support for hundreds of billions of objects, and second, a low latency requirement (~100 ms or lower) that is necessary to be able to support fine-grain mobility without disruption to application flows. We are currently investigating two alternative designs for such a service. The first approach proposes an in-network scheme where routers support an efficient 1-hop DHT (distributed hash table [8,9]) service to distribute GUID:NA mappings across the Internet [10]. The second approach uses a distributed overlay service with locality-aware replication for each name and partitioning across names so as to optimize latency while respecting capacity constraints. We describe below details of the first approach and illustrate the service API, which is common for both designs.

The router DHT approach scales by cooperatively and equably hosting NA:GUID mappings on routers of participating networks. A consistent hash function along with the network reachability information ob-

tained from an inter-domain routing protocol is used to partition and distribute the GUID:NA mappings among the networks. At a basic level, the service supports two operations: *insert* (or update) and *query* (or lookup). The insert operation stores in the GNRS the GUID:NA mapping of a network-attached object soon as it associates with a network, and the query operation resolves the set of NAs that a queried GUID is currently attached to. Figure 10 illustrates these two operations. As host A attaches to the network, it sends an insert message to the designated GNRS service router within the local network. The service router applies a predefined consistent hash function to the GUID to derive a value X, which is a valid network identifier in the global networks' namespace. The inter-domain routing tables are consulted to ensure participation and reachability of network X and the insert message is forwarded to the GNRS router in network X, which then stores the mapping. A query (looking for A's location) from a second host B follows similar steps by first sending a query message to its local GNRS router. The identical hashing scheme is used to forward the query to the network and GNRS router that holds the mapping for A's GUID to resolve the query. For reasons of performance, reliability and trust, the in-router DHT scheme normally employs more than one ($k > 1$) hash function at a time to achieve k replicas for each mapping.

In a proof-of-concept instantiation of the router DHT scheme, we employed SHA-1 based hash functions and used BGP as the inter-domain routing protocol to consult network participation and reachability. The network namespace here was the set of IPv4 prefixes announced by ASs. From the hash of the GUID (mapping into the network namespace) we obtain the network that should host the mapping. This configuration was evaluated by simulating an Internet-scale AS-level network topology (with ~26000 ASs) with actual inter-AS latencies obtained from the DIMES database [11]. Figure 11 shows the CDF of the query latency from 100,000 queries to GUIDs with mappings distributed equably across the ASs. The number of replicas, k, was varied between 1 and 5 to study the impact on the key performance metric of query response time. The effect of k can be clearly seen with the leftward shift of the response time CDF curves with increase in k. The 95th percentile of query latency for k=5 is below ~100ms (typical handover delay incurred in cellular systems). This is a promising result towards enabling real-time mobility at scale within the MobilityFirst architecture.

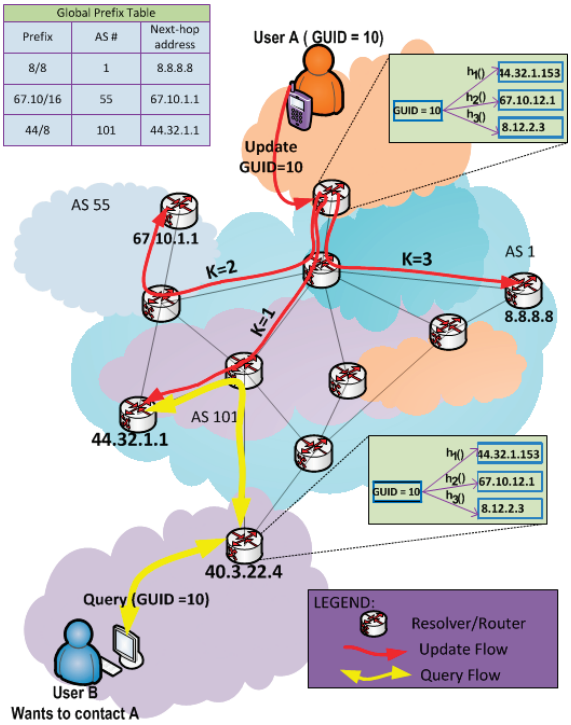

Fig. 10: Overview of GNRS protocol. The GUID:NA update from host A reaches multiple (k=3) server replicas; the query from host B (for A's GUID) is resolved from the nearest replica.

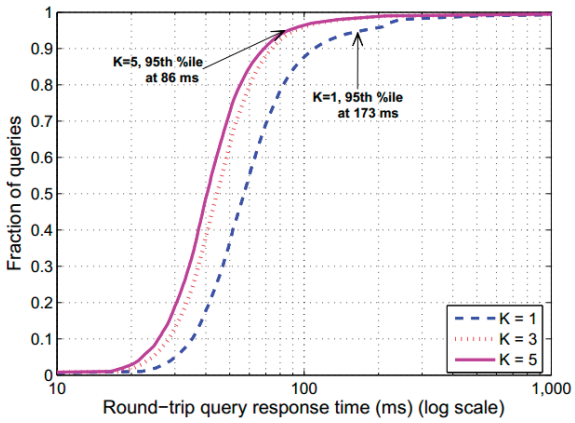

Fig. 11: GNRS query response time

3.2 GSTAR Routing: A major goal for our intra-domain routing protocol was for it to seamlessly span use in wired/wireless access networks including cellular, WiFi, vehicular, and extreme DTN scenarios. More specifically, the protocol targets the following features:

- Adapt to radio link quality fluctuations and congestion in the network

- Disruption-tolerance: handle disconnections and partitions in the network
- Leverage storage in routers to handle wireless and mobile challenges
- Work seamlessly over hybrid path segments of wired and wireless nodes

Prior work on cache-and-forward (CNF) networks introduced protocols capable of handling hop-by-hop transport over varying link qualities by allowing routers to make in-network decisions to temporarily store data packets [12,13]. Our approach is to extend CNF, giving it a higher degree of delay-tolerance which is necessary in many mobile edge networks. Furthermore, we have augmented the original CNF routing and transport approaches to allow for a more proactive backpressure-based congestion control mechanism, which is particularly important when data is being stored for long periods of time. We refer to this new protocol as *GSTAR* (*G*eneralized *S*torage-*A*ware *R*outing) [14].

GSTAR basically combines a link state protocol with DTN capabilities to support ad hoc, disconnected and partitioned network conditions. There are three types of control messages exchanged: (1) link probes, (2) flooded link state advertisements (F-LSA), and (3) epidemically disseminated link state advertisements (D-LSA). Link probes allow nodes to obtain both time-sensitive expected transmission time (ETT) values for adjacent links as well as a rough idea of the connectivity pattern with other nodes. F-LSAs allow nodes within the same partition as an advertiser to obtain short term ETT, long term ETT, and storage availability information about the advertiser and its adjacent links. D-LSAs allow all nodes, even those outside of an advertiser's partition, to obtain general connectivity information about the advertiser.

All nodes periodically probe for neighbors, making a note of which neighbors are currently available and what the ETT (directly computed) for the links are. Over time, they average the ETT values for a single link and compute a "long term ETT" value. All nodes in a partition periodically learn about the recent short term and long term ETTs and available storage via periodically flooded F-LSA messages. They also learn about general connectivity patterns for the entire network via D-LSA messages. Therefore, two graphs are created: (1) the intra-partition graph, where vertices are nodes within the partition and edges have both a short and long term ETT values associated with them, and (2) the inter-partition graph where vertices are nodes in the network and edge weights are a metric indicating the frequency or likelihood of two nodes being in contact.

When a PDU arrives or is sourced, the router first checks the intra-partition graph to see if the destination ID is a valid vertex. If it is, it will solely use that graph to route the data. In this case, it will obtain the shortest path *with enough available space* using the short term link ETTs along the first few hops and the long term link ETTs after. After obtaining a valid path, the router will then have to make a decision to forward to the next hop on that path, or store the data for later. This is done using a three-dimensional metric including: (1) short term ETT over the path, (2) long term ETT over the path, (3) exponentially weighted view of storage availability over the path. Proactive congestion control is built into the metric. Figure 12 illustrates the store vs. forward decision space.

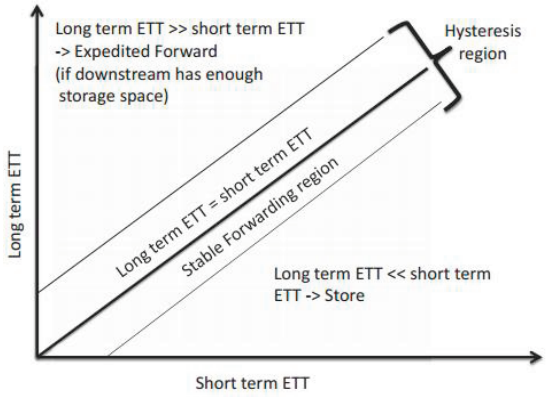

Figure 12: Store vs. Forward Decision Space

If the destination ID is not found in the intra-partition graph, then the DTN graph is consulted. A small set of non-overlapping shortest paths are computed. The goal is for a replica to make progress along each of these paths; therefore, the router must find, for each path, the furthest node on that path such that it is still within its intra-partition graph and has enough available storage. Replicas are transmitted to these nodes using the intra-partition graph technique, and then stored there until the appropriate next hop is met. An illustration of how data progresses through a domain is found in Fig. 13.

Simulation and prototyping results obtained on GSTAR so far [15] indicate that the store/forward decision making process allowed for a more robust, higher throughput protocol compared to current

techniques - see for example, the ns3 simulation based plot in Figure 14 that shows goodput gains by GSTAR over storage-augmented link-state (both using hop-by-hop transport); note that gains relative to conventional TCP/IP are much higher. Similar performance gains have been observed with ORBIT testbed [20] experiments with ~10-20 node access networks with end-point mobility and varying channel quality.

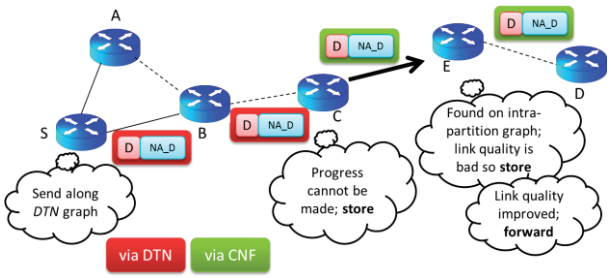

Fig. 13: Data progressing through a domain using router storage and DTN routing capabilities

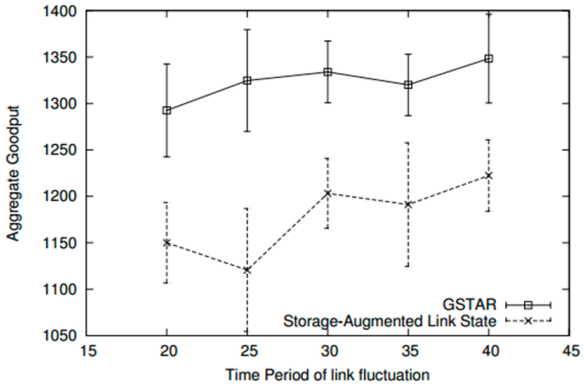

Fig. 14: NS3 Simulation Results showing GSTAR Performance vs. Basic Hop-by-Hop/OLSR

To conclude this section, we note that an additional key component of the architecture is the inter-domain routing protocol currently under investigation. The inter-domain protocol in MobilityFirst is different from BGP because of the flat network address structure (i.e. no prefix) and the need to expose some edge network quality information in order to support multi-homing and multi-path. In addition, the use of GUIDs opens up the possibility of late binding techniques in which the network addresses for delivery can be dynamically updated to reflect disconnections or mobility. An approach called edge-aware inter-domain routing is currently under investigation along with an alternative two-tier hierarchical scheme, and will be reported on in future work.

4. MobilityFirst Prototype on GENI

An early proof-of-concept prototype of the MobilityFirst architecture is currently under development, and was first shown at the GENI Engineering Conference-12, Kansas City in Nov 2011. This initial prototype is software-based, and the router elements are built as Click modular router [17] modules with GNRS, storage-aware routing (GSTAR), and hop-by-hop data transfer protocols implemented.

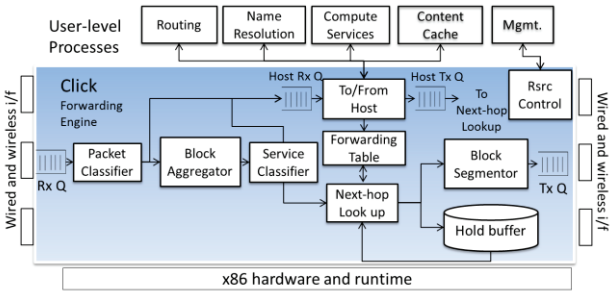

Fig. 15: Click-based Implementation of MF Router

The MobilityFirst protocol stack has also been implemented for Linux and Android platforms to enable client access. The stack includes the GUID service layer and a new set of service APIs that enable GUID-based message delivery (unicast, multicast, DTN, etc) and new GUID-based content retrieval and context addressing services . The Click-based implementation along with additional service modules is outlined in Figure 15. It shows a two-level abstraction with fast-path forwarding and block storage implemented as Click elements, and slow-path processing, such as GUID resolution, routing control and management, handled by user-level processes. Store or forward decisions using GSTAR are taken at the level of a PDU after all data packets for the PDU have been received from upstream node. However, prior to a forwarding decision an SID-based classifier determines if any additional processing on the PDU such as content caching or compute-plane services are applicable. Packets requiring a slow-path service are handed to the appropriate user-level module on the host. The classifier also sets aside PDUs that need a GNRS resolution. Such PDUs are buffered until corresponding NA(s) have been retrieved through the name resolution service process. Finally, the next-hop lookup determines if a packet is to be

forwarded or stored temporarily due to transient path quality issues or disconnections. PDUs ready to be forwarded are segmented and transmitted, while PDUs in hold buffer are re-visited periodically to force a forwarding action.

The experimental system prototyped and demonstrated at GEC-12 in Kansas City consisted of seven MobilityFirst core and access routers spread across the US. The core connected two wireless edge networks located at BBN, Cambridge, MA and WINLAB, Rutgers, North Brunswick, NJ, which supported both WiMAX BSS and WiFi access points for mobile host access. The network topology is illustrated in Figure 16, and shows access and core components as 7 networks. Each of the networks host a GNRS enabled router, and together form the distributed GNRS service plane.

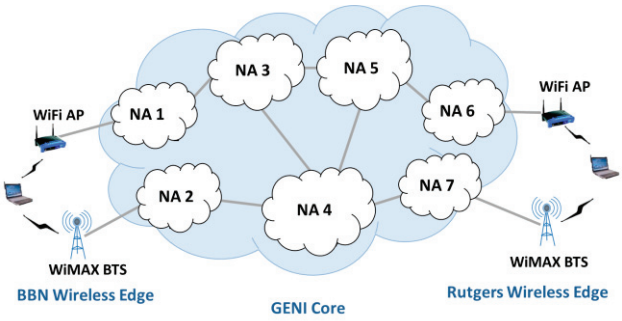

Fig. 16: GENI proof-of-concept demonstration topology

The corresponding physical topology of the GENI setup is shown in Figure 17. This includes OpenFlow switches [18] and ProtoGENI hosts [19] located at various sites across the US connected by two backbone networks, Internet2 (I2) and National Lambda Rail (NLR). Each ProtoGENI node ran MobilityFirst router prototype and network services, while the OpenFlow swtiches established layer-2 connectivity across the nodes. The paths from WiFi and WiMAX access networks are separated until they reach the core by placing their traffic in separate VLANs (3715, 3716), which are then bridged at the core (at a node in the Clemson site). We used Linux laptops connected to WiMAX and WiFi access networks to run MobilityFirst host stacks as network clients. Overall, the configuration offered realistic RTT delays between routers and also variety in link speeds and access techologies for end-hosts. In the demonstration scenario, the two dual-homed mobile devices at BBN and Rutgers networks served as a content server and subscriber.

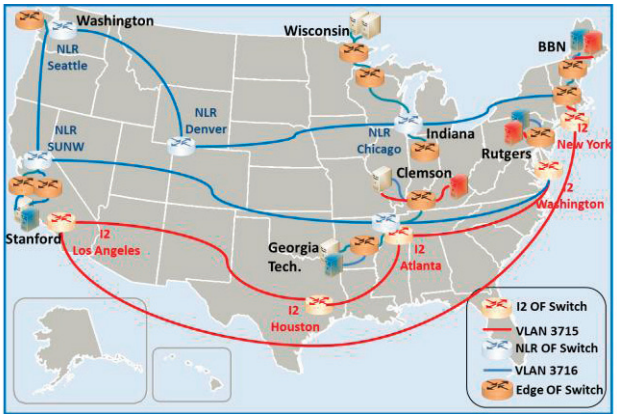

Fig. 17: Physical network configuration during GENI proof of-concept demonstration

On the above network, we demonstrated a content delivery application with one of the two dual-homed mobiles acting as content server (GUID 101) and the other a subscriber (GUID 201). A labeled content is requested by the subscriber through a *get(GUID)* message (see Figure 18). The content delivery operation involves first a GNRS resolution for the subscriber's NAs followed by GSTAR routing of the protocol data unit (a media file) on a hop-by-hop basis to the two interfaces of the multi-homed subscriber.

The experiment also demonstrated GSTAR protocol's capability of dealing with edge network bandwidth variation and occasional disconnection which are normal in the WiMAX/WiFi edge networks under host mobility. MobilityFirst routers stored data blocks until channel conditions improved sufficiently for forwarding to resume along the path. GSTAR automatically chose one (best) or both multi-homed paths when available. Note from the figure that data packets headed to the subscriber are initially resolved to both of the mobile's network associations - NA6 and NA7. As shown, a bifurcation point that sees both paths to the multi-homed client then chooses a suitable path (may even stripe across both). When associations dissappear due to mobility, GSTAR performs a late-binding with a fresh GNRS lookup to retrieve the latest NA of the mobile for a renewed delivery attempt.

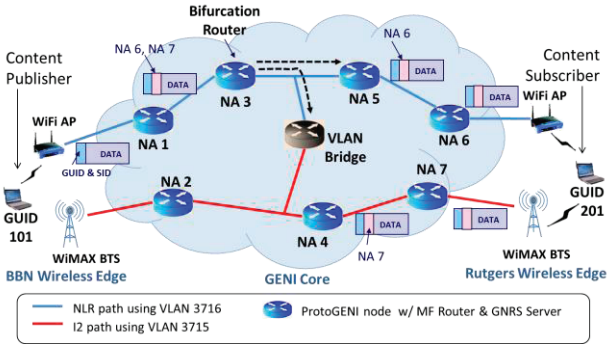

Fig. 18: Content delivery application demonstrated on GENI deployment of the MobilityFirst prototype router and services

5. Concluding Remarks

In this paper, we have presented an overview of the *MobilityFirst* future Internet architecture currently under development under the NSF FIA program. The design includes several key concepts for mobility-centric networks including clean separation of name from routable address, global name resolution service (GNRS), generalized storage-aware routing (GSTAR), routing layer support for multi-path, multicast and multi-homing, and content- or context-aware message delivery. A proof-of-concept prototype for the MobilityFirst protocol stack has been successfully developed on the GENI experimental testbed and first demonstrated in Nov 2011. The results so far are promising, and we expect to report more definitive results with protocol details and performance numbers over the next 1-2 years. Further work will include inter-domain routing aspects, designs for content- and context-aware services, management plane features, and computing layer services.

5. References

[1] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2011-2016.
[2] D. Raychaudhuri and M. Gerla, "New Architectures and Disruptive Technologies for the Future Internet: The Wireless, Mobile and Sensor Network Perspective", Report of NSF Wireless Mobile Planning Group (WMPG) Workshop, Aug. 2005. http://www.winlab.rutgers.edu/WMPG
[3] M. Sems. Debate rages over who should control ICANN. Processor 31(16):7, Jun. 2009.
[4] X. Liu, X.Yang, and Y. Lu. To Filter or to Authorize: Network-Layer DoS Defense Against Multimillion-node Botnets. ACM SIGCOMM, Aug. 2008.
[5] D. Andersen., H. Balakrishnan, N. Feamster, T. Koponen, D. Moon, and S. Shenker. Accountable Internet Protocol (AIP). ACM SIGCOMM, Aug. 2008.
[6] R. Moskowitz and P. Nikander. Host Identity Protocol (HIP) Architecture. IETF RFC 4423, May 2006.
[7] M. Li, D. Agrawal, D. Ganesan and A. Venkataramani. Block-Switching: A New Paradigm for Wireless Transport. USENIX NSDI 2009.
[8] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications. IEEE/ACM Trans. on Networking, Feb. 2003.
[9] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content-Addressable Network. ACM SIGCOMM, Aug. 2001.
[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. IEEE ICDCS, Jun. 2012.
[11] The DIMES Project. http://www.netdimes.org
[12] S. Paul, R. Yates, D. Raychaudhuri and J. Kurose. The Cache-And-Forward Network Architecture for Efficient Mobile Content Delivery Services in the Future Internet. First ITU-T Kaleidoscope Academic Conference on Innovations in NGN: Future Network and Services, May 2008.
[13] S. Gopinath, S. Jain, S. Makharia and D. Raychaudhuri. An Experimental Study of the Cache-and-Forward Network Architecture in Multi-hop Wireless Scenarios. Local and Metropolitan Area Networks (LANMAN), Apr. 2010.
[14] S. Nelson, G. Bhanage, and D. Raychaudhuri. GSTAR: Generalized Storage-Aware Routing for MobilityFirst in the Future Internet Architecture. ACM MobiArch, Jun. 2011.
[15] N. Somani, A. Chanda, S. C. Nelson and D. Raychaudhuri. Storage Aware Routing Protocol for Robust and Efficient Services in the Future Mobile Internet. IEEE FutureNet V Workshop, Jun. 2012.
[16] Global Environment for Networking Innovations (GENI). http://www.geni.net.
[17] Click Modular Router Project. http://read.cs.ucla.edu/click/
[18] OpenFlow Switching Spec. http://www.openflow.org/
[19] ProtoGENI. http://www.protogeni.net/trac/protogeni
[20] ORBIT Wireless Network Testbed. http://orbit-lab.org
[21] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang and J. Wang. A First Look at Cellular Machine-to-Machine Traffic -- Large Scale Measurement and Characterization. SIGMETRICS / Performance, Jun. 2012.