An Architecture for Tetherless Communication

A. Seth, P. Darragh, S. Liang, Y.Lin, and S. Keshav
School of Computer Science, University of Waterloo
Waterloo, ON, Canada, N2L 3G1

Abstract— **In the emerging paradigm of tetherless computing, client applications running on small, inexpensive, and smart mobile devices maintain opportunistic wireless connectivity with back-end services running on centralized computers, enabling novel classes of applications. These applications require a communications infrastructure that is mobility-aware, disconnection-resilient and provides support for an *opportunistic* style of communication. It should even be able to function across network partitions that might arise when end-to-end communication is not possible. We outline, design, and evaluate the implementation of an architecture that provides this functionality. We show that it is possible for next-generation mobile devices to obtain up to 80-fold improvements over conventional mechanisms by exploiting opportunistic WiFi links, and that this benefit can be delivered as an overlay that is compatible with the current Internet.**

I. Introduction

In the emerging paradigm of tetherless computing, client applications running on small, inexpensive, and intelligent mobile devices maintain opportunistic wireless connectivity with back-end services running on centralized computers, enabling novel classes of applications that can address problems ranging from rural development, to environmental monitoring, healthcare and education. For instance:

- A bus carrying an 802.11 access point can wirelessly pick up data as it drives past rural areas that are far from Internet connectivity [8, 16, 18]. Later, the bus can forward the data to the Internet when it passes an Internet-connected wireless hot spot. Symmetrically, the bus may also deliver data to an otherwise disconnected endpoint as it drives past.
- A health care provider can record *vitals* and test results of home-care patients in a mobile device. When driving past an access point, this information can be relayed to a central database for archiving and also sent to the attending physician for follow-up diagnosis. Symmetrically, instructions for patient care can be given to the provider during this opportunistic contact.
- Solar- or wind-powered sensors in remote areas can collect and store data at a base station, and this data can be picked up by vehicle driving past, or an aircraft flying overhead. This would be both efficient and cost-effective, for instance, in monitoring ground-water or stream contamination in remote or rural watersheds.

In these examples, an edge application client opportunistically and wirelessly communicates with a mobile router (also called a 'ferry' [29] or 'data mule' [24]) that carries data to and from a centralized server. Such tetherless applications require us to address several problems:

- *Addressing*: During an opportunistic contact, a device is usually addressed by a DHCP-assigned private IP address. How should a sender address a receiver whose address keeps changing? Today's solutions are to either use a Mobile-IP address [19] or a HIP address [15]. However, as we demonstrate later, neither solution works well when the mobile device is mostly disconnected and only opportunistically connected, or when the network is partitioned.
- *Location*: A closely related issue is locating a device that moves from one region to another. Essentially, every time the mobile moves, it must update its location in a well-known table. This is located at the Home Agent for Mobile IP, and at the Home Location Register in cell phone networks, or in a set of distributed home agents in DHARMA [12]. However, this does not address the problem of how a disconnected client in a remote partitioned network should update this centralized location table.
- *Routing*: Given a mobile device's location, how should the infrastructure determine a route, and if there is a choice, the best route, to the device? For instance, if there are are two buses which go past a rural dispensary, which one should be used to reach it? The problem of finding routes in networks with periodically disconnected (that is, scheduled) and opportunistic links is both challenging [9] and open. In this paper we present a simple and pragmatic solution that scales reasonably well, but can be fragile. However, it is sufficient for us to demonstrate the validity of our architecture that may be proposed for such networks.
- *Session persistence*: From the perspective of an application developer, it is convenient to provide the abstraction of a session that persists despite disconnections, mobility, and changes in the underlying wireless technology (such as Bluetooth, GPRS, or WiFi). Note that TCP connections may restart each time a mobile disconnects and reconnects, and will certainly terminate on reassignment of IP address. How then can a client and server maintain communication progress despite disconnections and partitioned networks? Solutions such as TCP Migrate [25] and Rocks and Racks

[28] are only partial solutions because they are end-to-end, and therefore extend only from the mobile device to a data mule. Instead, we require solutions that extend session persistence from the device all the way to the server. Our solution for session persistence extends the approach pioneered by PCMP [17] and is described in more detail in Section III E and [23].

Based on these considerations, we enumerate some essential goals for a tetherless communications architecture:

1. *Mobility transparency*: It should always be possible to address, locate, and route to a mobile device.
2. *Disconnection transparency*: End-to-end communication state should persist across disconnection periods. Moreover, simultaneous presence of both ends of a (transport-layer) end-to-end link should not be necessary.
3. *Low control overhead*: The architecture should maximize the usage of communication opportunities by minimizing control overheads and locating data as close to an opportunistic link as possible.
4. *Internet-compatible*: This is required to allow deployment in the real world.
5. *Secure*: In addition, we would like communications to be secure; this is addressed in related work [22].

We present a solution that has these characteristics. We describe related work in Section II, followed by a description of the tetherless communication architecture in Section III. We then present an analysis for opportunistic communication in Section IV, and the simulation model and results in Section V and VI respectively. Our implementation efforts are listed in Section VII. In Section VIII we discuss some high level issues related to mobility and opportunistic communication, and finally present our conclusions and future work in Section IX.

II. Related Work

A. Cellular networks

Cellular telephone networks certainly seem to have solved the problem of tetherless communication, providing nearly seamless voice communication despite mobility and transient disconnections. However, cellular telephony (and data over cellular links) is low-bandwidth on the order of 100kbps, even with 3G, and expensive. Moreover, widespread penetration of cellular networks is limited to urban areas; rural and sparsely populated areas, especially in developing countries, cannot afford ubiquitous cellular coverage. For these reasons, existing cellular network solutions cannot be trivially retargeted for tetherless computing.

Compared to cellular networks, 802.11-based wireless networks provide higher throughputs and are cheaper to deploy, but they have complicated handoff and roaming issues that need to be handled due to the fact that 802.11 networks have a much smaller coverage area. The challenge in tetherless communication is to mimic the considerable capabilities of cellular networks by composing a large number of heterogeneously administered wireless LANs and potentially using a motorized backhaul. This would result in a network that would not support interactive communication, but would be far cheaper and have an order of magnitude larger capacity than cellular networks.

B. Network layer mobility solutions

Although schemes such as mobile IP [19], HIP [15] and I3 [27] provide mobility transparency (and with HIP, identity management), they cannot function effectively in partitioned networks. In particular, they do not address the problem of updating a location register when a mobile is able to access the Internet only through a proxy. Hierarchical solutions such as Cellular IP [5] and Hierarchical Mobile IP [26] move the update (or anchor) point closer to the mobile, but they still do not solve the problem of disconnection. Finally, currently known solutions that postulate globally unique IDs, such as I3 and HIP, do not allow a transiently connected mobile to resolve the ID of a recipient using only the resources available to the mobile or to the bus passing by. That is, DNS-like resolution of human-readable and/or location-independent names cannot be translated to location-specific identifiers *in the field*; they require the availability of a distributed resolution hierarchy.

C. Transport layer disconnection tolerance solutions

This includes protocols like TCP Migrate [25] and Rocks-and-Racks [28], which allow a TCP connection to resume on network reconnection. However, these protocols only support TCP, and also only on an end-to-end basis. Therefore they cannot function in a highly partitioned network where TCP connections can span only a part of the end-to-end path.

D. Email

Electronic mail has many of the characteristics we desire: disconnection tolerance, message semantics, and ubiquitous deployment. However, existing email systems do not support mobility: a mobile user is constrained to always pick up data from the *home* email server, no matter where it is located. Second, routing in email systems is done manually using MX records, which makes it both static and error prone. It would be far better to use automatic routing protocols that dynamically adapt to mobile users and changing network conditions. Finally, email suffers from the problem that sender identities are not verified, resulting in an enormous volume of ‘spam’. While not described in detail here, our solution allows verification of the identity of every sending entity using Identity Based Cryptography [22]. This alleviates the spam problem.

E. Delay Tolerant Networks (DTN)

Unmodified TCP cannot be used over challenged networks where there are frequent disconnections, or when the network regions to be traversed are heterogeneous and connectivity is not always available. DTN [8] instead proposes the notion of *bundle transfer*, where the data is wrapped into bundles (similar to email messages) and these bundles are transferred using an overlay network. DTN routers or *custodians* have a large persistent bundle store, where bundles await transfer to the next DTN router whenever connections become available. Custody transfer occurs as the bundles are transmitted across the network, and the responsibility of reliable delivery of each bundle is passed from one custodian DTN router to the next.

DTN addresses the problem of disconnections in a novel manner, even though it constrains applications to be non-interactive in nature. End-to-end connectivity is not necessary with DTN. Senders and receivers can inject and retrieve bundles from the DTN overlay according to their individual connectivity schedules. A side benefit of DTN, as we show in Section IV, is that high throughput rates can be obtained for opportunistic delivery if bundles are routed to DTN routers physically close to potential receivers.

However, the existing DTN architecture has no support for locating and routing to mobile devices, which is an essential requirement for tetherless computing. Moreover, it does not support persistent session state, nor does it address the issue of how a disconnected end node should discover the identifier associated with a remote, and potentially mobile, end point. We resolve these problems in our work.

III. Tetherless communication architecture

A. Overview and definitions

Based on the goals outlined in Section I, we believe that the following features must be included in *any* architecture that supports both mobility and disconnection:

1. *Intermediate persistent storage:* End to end connections will not be always possible for disconnected mobiles. Hence, intermediate infrastructure nodes should have a persistent storage capability where senders can inject data and receivers can pick up data according to their individual connectivity schedules.
2. *Lookup on a globally unique identifier(GUID):* Each mobile host should possess a GUID that maps to the current location of the mobile. The GUID should not change as the mobile moves and its mapping should be updated periodically with the new location.
3. *Forwarding to changing mobile locations:* The data forwarding function must incorporate a lookup step to map from the GUID to the current location.

It turns out that these three simple functionalities are sufficient to deal with both mobility and disconnection. We now explain the architecture in greater detail.

We address each end point with a GUID, and use distributed hash table (DHT) infrastructure such as OpenDHT [21] for looking up the location of a mobile based on its GUID. A DTN overlay network provides intermediate persistent storage, and data forwarding within the DTN network is described in Section III-D.

We first present some definitions.

1. *Region:* A region is collection of mutually reachable DTN routers, determined by administrative policies, communication protocols, naming conventions, or connection types. Regions usually are contained within a physical boundary, though a single region may span multiple geographical areas. We do not place any assumptions on the organization of regions.
2. *Gateways:* These are DTN routers with interfaces on more than one region.
3. *Custodians:* These DTN routers act as always-available proxies for intermittently connected mobile hosts. Custodians store data on behalf of disconnected mobile hosts and deliver them whenever the hosts reconnect to the network. We impose the simplifying constraint that custodians must not be mobile.
4. *Local DTN router:* This is the DTN router that communicates directly with a mobile. A local DTN router may or may not also be a custodian. A local DTN router may itself be mobile. For instance this models a bus that travels between villages with a DTN router on board.
5. *Near area:* This is defined as the set of wireless access points (APs) that are *closer* to a particular custodian DTN router than any other custodian. We do not define closeness precisely; one candidate would be the long-term mean RTT delay between an AP and the custodian DTN router.
6. *Near mobility:* Mobility within a near area is near mobility. Pre-authentication can used during near mobility to help reduce reconnection delays [14].
7. *Far mobility:* Mobility between near areas is far mobility. We assume that near mobility is much more common than far mobility and optimize our architecture for this case.

B. Location management

We identify all mobile hosts using an opaque globally unique identifier (GUID) that we will denote by I. Although our architecture does not require any specific semantics from the GUID, it turns out that choosing a node's GUID to be the MD5 hash (or some other collision resistant hash) of its human-readable identifier, such as the node-users's email address, allows the translation from a human-readable name to a fixed-length numeric GUID to be carried out without any additional infrastructure. This alleviates lookup problems such as those with I3, where a

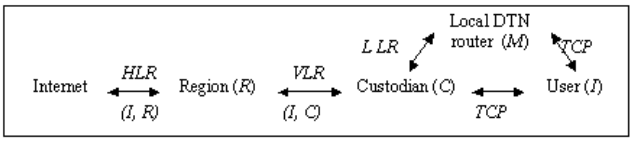

Fig. 1. Three stage hierarchy of lookups to support mobility

mobile needs to use the I3-DNS service to find its correspondent's I3 ID, and HIP, where a mobile needs to determine the cryptographically signed ID of its correspondent using PKI. This choice of GUIDs is also motivated by our security architecture [22]: essentially, the MD5 hash of a user is also that user's public key, and it can be given a private key by a Private Key Generator using Identity Based Cryptograpy [3].

The Internet region, which has special status in our architecture, maintains a lookup table which maps a mobile's GUID (I) to its current region (R). Although any lookup mechanism can be used, we use a Distributed Hash Table (DHT) to gain scalability and fault-resilience. Following cellular telephony terminology, we call this lookup table the Home Location Register (HLR). If a mobile is simultaneously present in multiple regions; for instance, when the mobile host is reachable on two different bus routes; the HLR resolves the GUID to multiple regions, both of which represent viable routes.

Each region maintains one or more Visitor Location Registers (VLR) that is either stored at, or accessible to, all of its gateways. The VLRs store a mapping from the GUIDs of all mobile hosts currently in the region to the custodian DTN router (C) for that mobile. Multiple mappings for different custodians are stored if the mobile can pick up bundles from more than one custodian.

Finally, each custodian maintains a Local Location Register (LLR) that maps from the GUID to the best last-hop fixed or mobile local DTN router (M) for each mobile. If the mobile picks up data directly from the custodian, without an intermediate local DTN router, this is reflected in the LLR.

This three-stage lookup hierarchy is shown in Fig. 1. When a mobile device moves, its location information is updated, if necessary, in zero or more location registers. If a mobile moves such that its local DTN router doesn't change, then none of the location registers are updated after the move. If a mobile moves such that its custodian doesn't change, but its local DTN router changes (i.e. a near move), then only the LLR at the custodian is updated to reflect the new best local DTN router for the mobile. If a mobile changes custodians within a region, the LLR at the new custodian is updated, and all VLRs in the region are updated to point to the new custodian. Finally, on a far move, the HLR, VLR, and LLR must all be updated both in the new and old regions. It is important the HLR only be modified after all the VLRs in the new region are aware of the mobile, as described in Section III-D. Presently we have implemented our own protocol for location management, but a protocol built on standard mechanisms like SIP based signalling is equally usable.

There are many ways of maintaining the location registers. If the region is large (like the Internet region) then the location register can be maintained in a DHT like OpenDHT [21], or in a central database. If the region is small then it can be maintained in a lookup table at each gateway and custodian.

C. Discovering local and custodian DTN routers

When a mobile moves, its location must be updated in the location registers. Therefore, the mobile device needs to determine its new region, custodian, and local DTN router. A local DTN router can always be queried for a list of *nearby* custodians and regions accessible through it. So, the problem reduces to that of a mobile finding a local DTN router after it has associated with a new wireless access point.

Note that when a mobile associates with a wireless access point, one of two cases must hold:

1. If the wireless access point is not itself connected to the Internet, then, in order for data to be ferried to the Internet, it must be co-resident with a mobile DTN router (as on a bus). The access point must therefore also be the local DTN router.
2. If the wireless access point is connected to the Internet, then the mobile can potentially look up a location service with location-specific information, such as the SSID of the access point, the current zip code, or its GPS location, to find a *close* local DTN router. Alternatively, when the access point is initially set up, this information can be hand-configured in the same way that it is configured with the address of a DNS server.

A local DTN router, when queried, may return more than one choice of custodian. Indeed, since the local DTN router may in fact be only one overlay hop away from *all* custodians on the Internet, the choice of custodian may be non trivial. We believe that a scheme similar to I3 trigger sampling [27] can be used to choose a *close* custodian. We are looking into algorithms for optimal choice of custodian in ongoing research. In any case, our scheme will work correctly with any custodian - choosing a *close* custodian is simply a performance optimization.

D. Routing

We now present the detailed routing protocols used in our architecture. The key concepts used in the design are late binding, default routing, reverse path forwarding, *make-then-break* updates, and group communication protocols as explained next.

As mentioned earlier, we give special status to the Internet region: this is the region that maintains a binding

from a mobile's GUID to its current region. We assume that every DTN router is manually configured with a default route that is its next hop to get to the Internet region. The next hop could be over a scheduled link and therefore may not always be available. We also define a special region name called *unbound*. When a DTN router gets a bundle addressed to the *unbound* region, it either forwards the bundle on its default route to the Internet region, or, if it has at least one interface on the Internet, looks up the destination's GUID in the HLR to rewrite the *unknown* region with the mobile's current region. Subsequently, the bundle is forwarded to that region's Internet gateway over one single overlay hop. The gateway then uses its local routing tables, set up using Reverse Path Forwarding, as described below, to route the bundle to the mobile.

The use of unbound regions is a form of late binding that allows a disconnected node to send a bundle to a destination knowing only its GUID; it does not have to query the HLR for the mobile's current region. Note that the current DTN architecture [6] already supports the notion of late binding. Here, the idea is that the administrative ID portion (or, to be precise, the namespace-specific portion) of the DTN address is bound to an actual next hop only at the destination region. We extend this notion to allow even a node's region to be late bound.

Before we describe how routing tables are set up, we observe that mobility intrinsically introduces race conditions. Bundles may be sent to a mobile's old region, or they may arrive to a gateway or custodian in the new region before it has heard of the mobile. To avoid race conditions, the location registers must be updated with care. The basic principle is to reliably update location information using a group communication protocol [2], *before* old information is deleted (*make-then-break*). This way, when a link is broken, a new path is known to exist.

We now outline a detailed algorithm for location update management in case of an inter-region far move (the case of a near move, and the case where there is a single chosen custodian or a single gateway in the region, is a subset of this case):

1. The mobile associates with a wireless AP and is given a local IP address, for instance using DHCP. It discovers its local DTN router using one of the techniques described in Section III-C.
2. The mobile tells its local DTN router, using a REGISTER message, that it has moved to its new region from its old region. The local router uses this to update its local table to reflect the fact that this mobile is now reachable through it.
3. The local DTN router informs the mobile of its choice of *nearby* custodians.
4. The mobile chooses one or more custodians and informs the local DTN router of its choice.
5. The local DTN router participates in a group communication protocol to update *all* the chosen custodians' LLRs to make itself as their next hop to get to the mobile. This effectively uses Reverse Path Forwarding to dynamically construct routing table entries for that GUID in the region.
6. When Step 5 terminates, one of the chosen custodians is elected to forward the REGISTER message to all the gateways in the region by participating in a group communication protocol with all the gateways in the region. The result of this is to update their VLRs so that the GUID of the mobile maps to its set of chosen custodians.
7. When Step 6 terminates, one of the gateways is elected to update the HLR to point the mobile's GUID to the mobile's new region.
8. Next this gateway participates in a group communication protocol with the set of gateways in the mobile's old region to update their VLRs so that these VLRs *unmap* the GUID by mapping the GUID to *undefined*.
9. When Step 8 terminates, the gateway reliably multicasts an UPDATE message to all the custodians in the old region. The custodians send any stored bundles to the mobile in the new region simply by rewriting the bundles' destination region as *unknown* and forwarding them on its default route. This will send them to the Internet gateway for that region, which looks up the HLR to determine the new region. Because this is the last step, group communication is not necessary.

This *make-then-break* approach has the interlocks necessary to prevent race conditions and provides eventually always consistent semantics. For instance, if a VLR points to an old custodian, bundles reaching the old custodian will either be stored and eventually forwarded (when the location update reaches the custodian), or the custodian will find the GUID to be unmapped, in which case the bundles will automatically be forwarded to the new custodian.

The use of reverse path forwarding has both its pros and cons. On the one hand, in the absence of a definitive solution to the DTN routing problem, it offers a simple way to set up the routing tables in a region. On the other hand, these tables are fragile: if a link were to break, or a DTN router were to fail, the protocol does not recover gracefully from this failure. The lack of fault tolerance can be handled in several ways. For instance, a mobile can periodically send a REGISTER message to refresh paths to it. Similarly, an Internet gateway can periodically flood a message that allows all the routers in its region to discover the default path to the Internet gateway. These protocols would limit outages to approximately one update period, which may be sufficient in practice.

Nevertheless, we believe that reverse path forwarding should be viewed only as a stop-gap measure. Once there is consensus on an acceptable DTN routing, then the only role of the REGISTER message would be to update the

location registers, and the actual routing can proceed using any agreed-upon DTN routing protocol.

E. Session state

Our architecture provides persistent session state on top of DTN using OCMP (Opportunistic Communication Management Protocol) [23], which can maintain session state across disconnections. OCMP also provides support for application-specific plugins to transparently communicate with legacy application servers in a delay tolerant manner. It supports different channels for control data and bulk data transfers, can multiplex traffic on different network interfaces, and also aggregates the traffic from multiple applications over a single transport layer connection. More details of OCMP are explained in [23]. OCMP extends PCMP [17] as follows:

1. Support for multiple NICs in parallel
2. Support for arbitrary transport protocols including UDP with erasure codes, TCP Migrate, and Rocks-and-Racks. PCMP is essentially TCP centric.
3. Integration with a database for session persistence across node restarts. This allows nodes to be powered down to save power.
4. Session state can be encapsulated and transferred from one OCMP proxy to another to maintain locality and choose the closest proxy for minimum proxy-to-endpoint RTT.
5. Servers can push data to OCMP proxies, and the data can then be picked up opportunistically by mobile devices.

F. Multiple home regions

Our architecture treats the Internet region as a special region that maintains the HLR. In fact, multiple regions can be designated as home-regions by embedding the ID of the home-region within the GUID of the mobile hosts. Thus, each mobile host can belong to its own home-region that maintains an HLR for it. Supporting this change requires a simple modification to the existing scheme. We assume that all HLRs are always available from the Internet region. Therefore, when an Internet gateway gets an unbound bundle, it simply looks up the mobile's current destination in the appropriate HLR. The rest of the scheme is unchanged. This allows each administrative region to manage its own HLR and GUID assignments.

G. Example

Fig. 2 illustrates the scheme in operation. The shaded region represents the Internet region. Region R1 is directly connected to the Internet, while Region R2 is connected through Region R1 to the Internet. (Note that R2 may have a scheduled link to R1 and so may never be able to directly access the Internet region.) The network contains custodians DTN-1 through DTN-3, and M-DTN represents a mobile DTN router, such as a bus.

Data transfer is illustrated from a fixed sender located in the Internet, through a custodian in the Internet, to a mobile receiver that changes its location from L-1 to L-5, through a series of movements.

IV. Analysis

We anticipate that most applications using tetherless communication will be for downloading bulk data from the Internet to a disconnected device. In this situation, we would like to determine the gain from using our architecture over other competing solutions. We consider a simple scenario of a connected sender sending data to a periodically disconnected and mobile receiver. The performance metrics we consider are the fraction of opportunistic connectivity intervals that a scheme can utilize and the amount of data a sender is able to transfer within the opportunistic connectivity intervals. The parameters in our analysis are as follows.

- R = Residence time of mobile receiver in a near-area, assumed to be exponentially distributed
- $\bar{R}$ = Mean residence time
- F_N = Number of connection periods in a near area before a far movement
- D_N = Delay needed for reconnection after a near move
- D_F = Delay needed for reconnection after a far move

We consider F_N pairs of (R_i, D_i) to denote F_N connection periods in a near area, with exponentially distributed residence times of R_i and connection establishment delays of D_i as follows:

$D_1 = D_F$

$D_i = D_N$ for $1 < i \leq F_N$

For a single connection interval indexed by i: q_i = opportunistic utilization ($0 < q_i < 1$) is defined as the fraction of the connection interval that can be used for data transfer, and is given by:

$q_i = 1 - D_i/R_i$ when $R_i > D_i$

$q_i = 0$ when $R_i \leq D_i$

Therefore:

P($q_i = 0$) = P($R_i \leq D_i$) = $1 - e^{-D_i/\bar{R}}$

P(q_i st. $q_i > 0$) = P($R_i > D_i$) = $e^{-D_i/\bar{R}}/\bar{R}$

In the special case of only near mobility (i.e no far mobility).

$\bar{q_i}$ = expected value of $q_i = \int_{D_i}^{\infty}(1 - D_i/x)e^{-D_i/\bar{R}}/\bar{R}dx$

Averaging over F_N periods, E($\bar{q_i}$) = $(\sum \bar{q_i})/F_N$

We then calculate the lim E($\bar{q_i}$)) as $F_N \rightarrow \infty$, to give the expected opportunistic utilization assuming that only near mobility occurs. We rely on simulations presented in

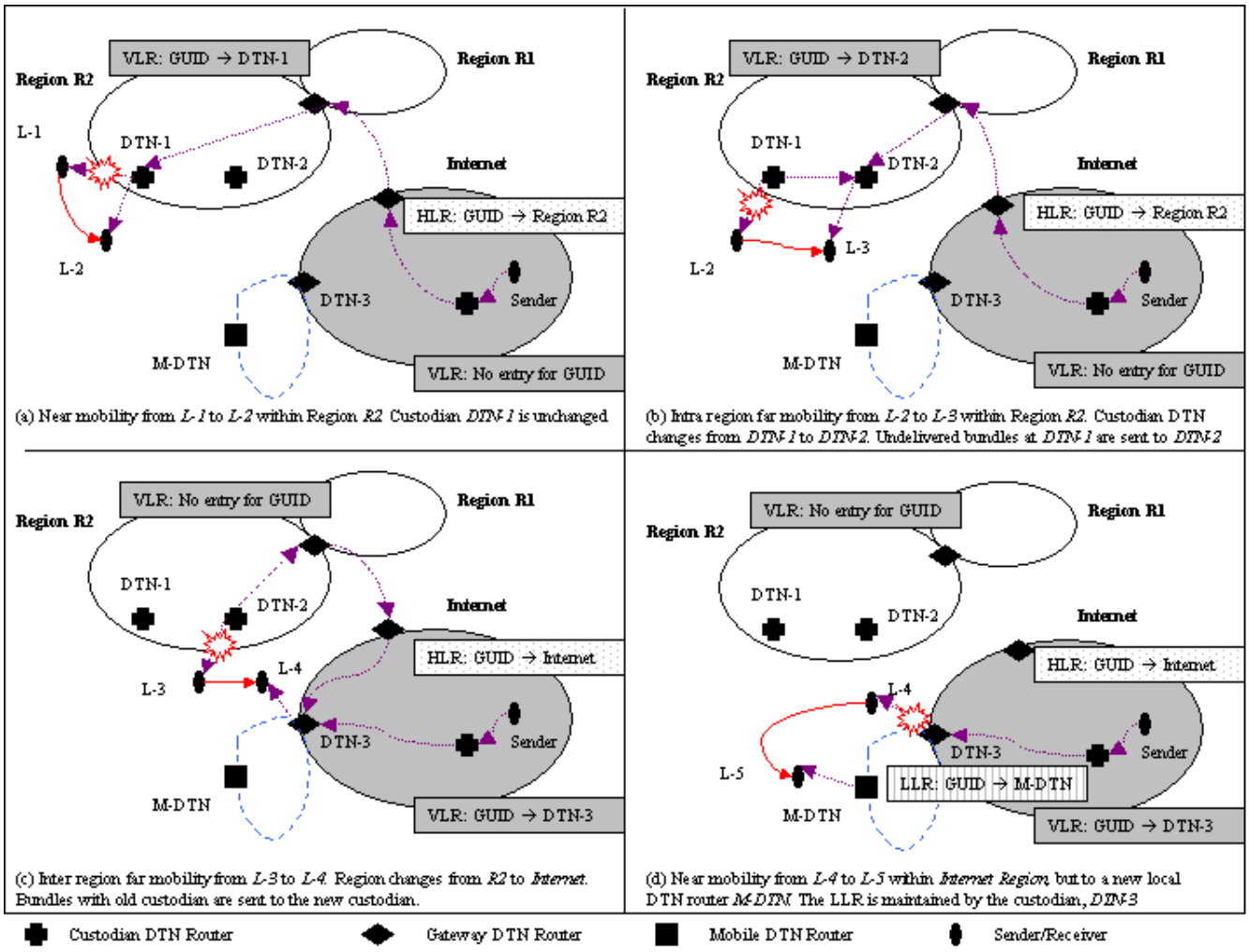

Fig. 2. TCA example

Section VI for studying the general case of both near and far mobility.

In order to calculate the amount of data transferred opportunistically during a connection interval, we use the TCP throughput equation in [11] to compute the expected throughput (μ) of a TCP connection from its RTT:

$$\mu = \frac{1.22MSS}{RTT\sqrt{L}}$$

Here, MSS is the mean segment size of the TCP packets, RTT is the round trip time, and L is the loss rate. So, the amount of data transferred per connection opportunity is:

$$\lim \mathrm{E}(\bar{q}_i) \,.\, \mu \,.\, \bar{R}$$

Using the above derivations, we now analytically compare TCA with end-to-end Rocks-and-Racks and end-to-end TCP on HMIP. In TCA, data is sent as bundles, with a TCP connection between the sender and the local DTN router, as well as between a mobile and a local DTN router. In the other two cases, we consider an end-to-end TCP connection over Rocks-and-Racks or HMIP. Recall that, unlike HMIP, Rocks-and-Racks permits limited disconnection.

A simple mean value analysis is done for estimating the delay (D_N and D_F) and RTT parameters of the different schemes. The connection establishment steps for the different schemes with both near as well as far mobility are enumerated below. Data is assumed to be piggy-backed on TCP Syn/Syn-Ack packets so that an extra RTT delay is avoided.

A. HMIP

Near reconnection

1. 802.11 reconnection and authentication
2. Update CoA at local HMIP MAP node and receive acknowledgement of update
3. Send reconnection notification to sender
4. Receive packet sent by sender

Far reconnection

1. 802.11 reconnection and authentication
2. Find out local HMIP MAP node
3. Insert CoA at local MAP node and receive acknowledgement of insert

Scheme	Delays		
$d_n = 15ms, d_f = 150ms, a_n = 50ms, a_f = 800ms$			
TCA	$D_N = a_n + 2d_n = 80ms$	$D_F = a_f + 5d_f + d_n = 1565ms$	$RTT = 2d_n = 30ms$
Rocks and Racks	$D_N = a_n + 2d_f = 380ms$	$D_F = a_f + 2d_f = 1100ms$	$RTT = 2d_f = 300ms$
HMIP	$D_N = a_n + 2d_f + 2d_n = 360ms$	$D_F = a_f + 4d_f + 2d_n = 1430ms$	$RTT = 2d_f + 2d_n = 330ms$

TABLE I
CONNECTION ESTABLISHMENT DELAYS

4. Update public address at home HMIP MAP node and receive acknowledgement of update
5. Send reconnection notification to sender
6. Receive packet sent by sender

B. Rocks and Racks

Near reconnection

1. 802.11 reconnection and authentication
2. Reconnection notification to sender
3. Receive packet sent by sender

Far reconnection

1. 802.11 reconnection and authentication
2. Reconnection notification to sender
3. Receive packet sent by sender

C. TCA

Near reconnection

1. 802.11 reconnection and authentication
2. Reconnection notification to local DTN router
3. Receive packet sent by DTN router

Far reconnection

1. 802.11 reconnection and authentication
2. Find out local DTN router
3. Update public trigger at far DHT node and receive acknowledgement of update
4. DHT node sends callback to old DTN router
5. Old DTN router sends data to new DTN router
6. Receive packet sent by new DTN router

Due to 802.11 pre-authentication, we assume that 802.11 reconnection and authentication delays are substantially lower in near mobility than in far mobility. Delays for locating local DTN and HMIP nodes, are neglected because the node discovery information is cacheable and can be provided by the APs themselves.

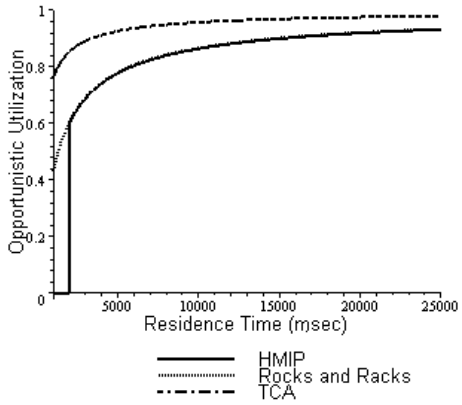

Fig. 3. Opportunistic utilization (E($\bar{q_i}$) Vs $\bar{R}$)

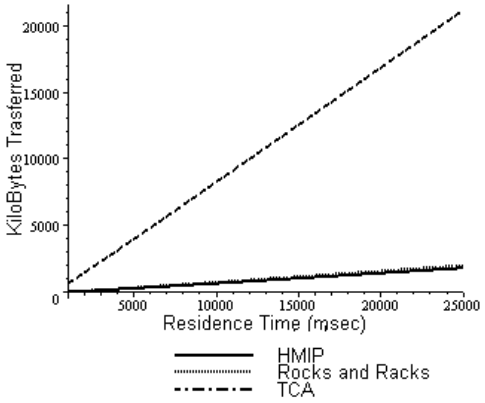

Fig. 4. Opportunistic data transfer ($\mu E(\bar{q_i})\bar{R}$ Vs $\bar{R}$)

Values of D_N and D_F are estimated by adding together representative values for each delay component constituting the connection re-establishment phases. We assume all delays between any two nodes within an AS (= d_n) to be equal to each other. Similarly, the delays between any two nodes in different AS (= d_f) are assumed to be equal, and an order of magnitude higher than the near delays. A representative value is chosen for authentication and 802.11 connection establishment delays with near mobility (= a_n), which is substantially lower than the delays with far mobility (= a_f) because it involves lesser and faster roundtrips for TLS like authentication and encryption. Values of RTTs for the different schemes are also estimated similarly. Table I shows the expected values of near and far reconnection delays and RTTs for the different schemes.

Fig. 3 and Fig. 4 show MAPLE [30] plots of the analysis equations derived in this section, using delay and RTT values shown in Table I. The MSS is fixed at 1500 bytes, and the loss rate is assumed to be 0.01

The results show that TCA gives the best performance, followed by Rocks-and-Racks, and then by HMIP. This is expected because TCA is able to reduce delay latencies in the case of near mobility, but end-to-end Rocks-and-Racks does not consider near mobility as being different from far

mobility, and HMIP is not able to utilize short connection opportunities until when the intervals are large enough to allow at least one complete TCP file transfer.

According to Fig. 3, all the architectures give comparable opportunistic utilization for large values of residence times. However, as shown in Fig. 4, the amount of data that the schemes are able to transfer is quite different. This is because the end-to-end throughput of a connection is inversely proportional to the RTT, and the RTT for Rocks-and-Racks and HMIP is almost an order of magnitude higher than the RTT for TCA. This is because TCA relocates the data closer to the mobile user. TCA is able to provide an order of magnitude performance increase over other mechanisms.

Number of nodes	5000
Number of AS	100
Average nodes per AS	50
AS constituting core network	35
Simulation duration	3 hours
TCP packet MSS	1500 bytes
Loss rate	0.01
Density of DTN/HMIP nodes per AS	1.0
Idle time during near movements	5 sec
Idle time during far movements	60 sec

TABLE II
SIMULATION PARAMETERS

V. SIMULATION

We analyze the behavior of TCA and compare it with end-to-end HMIP and Rocks and Racks by simulating the same scenario as that in Section IV, that is, of a fixed sender in the Internet sending data to a receiver that intermittently connects to the Internet.

A. Flow-level simulation

We use packet-level simulation only for control packets. Data packets are modeled as flows. Our simulation topology generator creates networks where each link is associated with a capacity and a delay (see below for details). We can therefore compute the bottleneck capacity as the smallest link capacity on any path. For simplicity, we ignore all cross traffic, so the bottleneck link capacity is identical to the path's available bandwidth.

The amount of data transferred in a given time period over UDP or TCP over Rocks-and-Racks is obtained by calculating the number of packets delivered at the available bandwidth on that end-to-end path. For TCP data transfer over HMIP we model disconnection and restart as follows: we compute the time taken to transfer a file of a given size over a path with a given available bandwidth. If this time exceeds the opportunistic connection time, we regard the transfer as aborted and model this as zero data transfer.

B. Network topology

We generate realistic network topologies using Brite [13], and then overlay this topology with DTN, HMIP, and sender nodes. This is done by randomly and uniformly selecting nodes to take on these roles, using as configuration parameters the density of DTN routers, density of HMIP nodes, and the density of senders/receivers within an Autonomous System (AS). We then determine the near areas corresponding to each DTN router.

C. Delay distribution

There is evidence showing that end-to-end delay distribution for Internet packets is a Gamma distribution with a heavy tail [4]. For our simulations, we use the same model with parameters as those given in [4].

D. File size distribution

Although file size distributions for email traffic is more suited to our application scenarios, due to the unavailability of such data, we use the distributions of Internet web traffic given in [1] for generating Pareto distributions for the sizes of files being transmitted.

E. Host mobility

We model the mobility of each mobile host as a continuous time Markov chain. Different states are kept for connection establishment intervals, connected intervals, idle interval spent in a near movement, and idle interval spent in a far movement. A supplementary model is used to simulate actual geographical movement of the mobile host, explained in greater detail in [10].

VI. SIMULATION RESULTS

We conduct two sets of experiments for measuring the performance metrics of the different architectures, with a single sender and a single receiver. For all our experiments, we use the parameters shown in Table II. All plots in the simulation graphs are an average of 30 runs. The measured standard deviations were very small (less than 5% of the mean values); for clarity, we do not show them in the simulation results.

A. Opportunistic utilization vs. mean residence times

Since our analysis covers the cases where no far mobility occurs, we use simulations to study the performance of the system in the presence of far movements. We choose a probability of near movement of 0.8 for these experiments. The results, shown in Fig. 5 closely correspond to the analysis results in Fig. 3. Similarly, Fig. 6 and Fig. 4 for the amounts of opportunistic data transferred also exhibit similar trends.

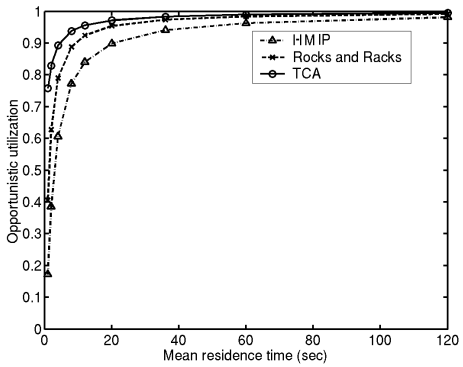

Fig. 5. Fixed prob of near mobility: Utilization Vs Mean residence time

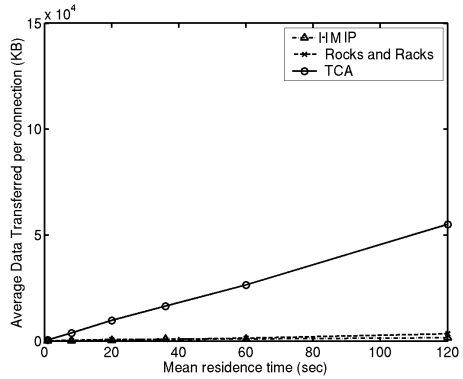

Fig. 6. Fixed prob of near mobility: Data transferred Vs Mean residence time

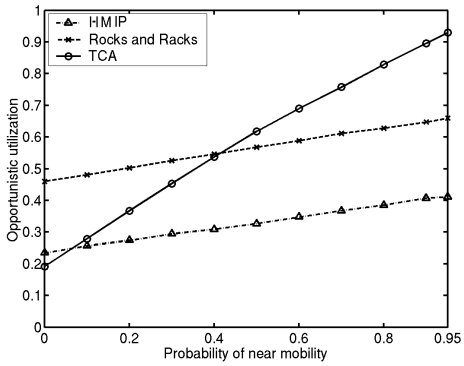

Fig. 7. Fixed mean residence time: Utilization Vs Prob of near mobility

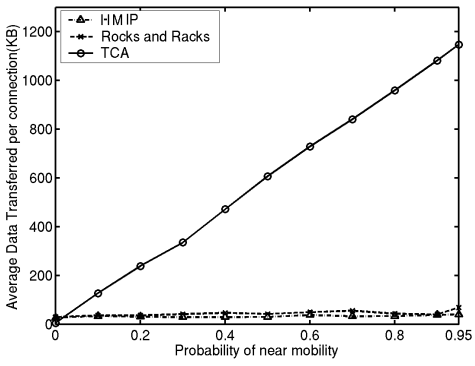

Fig. 8. Fixed mean residence time: Data transferred Vs Prob of near mobility

We observe from Fig. 5 that the smaller the residence times, the greater the advantage of TCA over the other schemes because TCA is able to utilize opportunistic connections better. From Fig. 6, It can also be expected that TCA will perform more efficiently with increasing DTN node density because the greater the node density, the smaller will be the RTT from the local DTN router to the mobile receiver. With the configuration parameters of Table II, we are able to realize an 80-fold improvement with TCA over the other schemes of Rocks-and-Racks and HMIP.

B. Opportunistic utilization vs. probabilities of near mobility

We next fix the mean residence time, and study the effect of near mobility on performance. The probability of near vs. far movement is varied from 0 to 1, implying a variation in the average number of connection periods (FN) from 1 to 8. The performance of TCA becomes better with increasing probabilities of near movements, because it is able to use near mobility more efficiently.

We see that a crossover point exists in Fig. 7 when the opportunistic utilization of TCA overtakes that of Rocks-and-Racks. This is because TCA encounters greater control overheads in far movements than the other schemes (due to its need to update both DHT entries and the old local DTN router), but as the amount of far mobility reduces, TCA performs much better due to more efficient connection utilization.

Fig. 8 shows the mean amount of data transferred during a communication opportunity. Unlike Fig. 7, no crossover is noticed in this case. This shows that even though TCA has a lower opportunistic utilization metric than the other schemes when there is a lot of far movement, the actual amount of data transferred is always more than with the other schemes because of reduced RTTs resulting from relocating the data to local DTN routers. In other words, the relocation of data to the local DTN router more than compensates for the extra control overhead during a far movement. This advantage will persist as long as end-to-end round trip times traversing the Internet backbone are about an order of magnitude higher than end-to-end round trip times within a near area. As speed-of-light propagation delays are already the dominant delay term in today's high speed backbones, this is likely to be the case indefinitely.

VII. Implementation

We have implemented a subset of our architecture on a testbed at the University of Waterloo. Our testbed is based on the DTN2 Reference Implementation from Intel Research, Berkeley [7]. We have made the following modifications to the reference implementation:

- We have modified the 'bundle router' object to allow default routes to the Internet region to be entered in the

configuration file.

- The bundle router scans an incoming bundle for three types of headers, those that have a REGISTER bit, those that have a FORWARD bit, and those that are unbound.
- On receiving an unbound bundle, if the bundle router is not an Internet gateway, it forwards the bundle on its default route. Otherwise, it looks up the DHT using the mobile's GUID as the key to determine the mobile's current region, and the Internet gateways for that region. The bundle header is then modified to bind the region in the destination address, and the bundle is sent to (one of) the destination region's Internet gateways.
- On receiving a REGISTER message, the local routing table is updated and the message is forwarded on the default route upstream to the Internet gateway. At the Internet gateway, the REGISTER message is used to update the VLR, and to update the OpenDHT Distributed Hash Table [21] running on Planetlab [20]. Before updating the DHT, however, the previous region of the mobile is looked up, the REGISTER message is converted to a FORWARD message, and sent to the Internet gateway of the previous region.
- On receiving a FORWARD message, the local routing table is updated to remove entries for that mobile's GUID. Also, all bundles in local storage with the mobile's GUID are modified to convert the region identifier to 'unbound', and these bundles are sent on the default route. The net effect is that after registration, unbound bundles from the old region eventually arrive at the old region's Internet gateway, where they are bound to the proper region and delivered. This allows a mobile's session state to be persistent across disconnections.

We ran into a problem with OpenDHT when using it as a persistent store. To prevent misbehavior, OpenDHT automatically deletes entries after a timeout period. Therefore, all GUID to region mappings need to be periodically refreshed. We make the Internet gateway responsible for refreshing the mappings. The algorithm to refresh the mapping is as follows:

1. Keep a list of GUIDs whose state needs to be refreshed. Add to this list when a REGISTER message is received and delete from the list when a FORWARD message is received.
2. Set a timer for the refresh interval. When the timer expires, for each GUID in the list, read its current region value. If the region matches the local region, refresh the entry; otherwise do nothing

The second step is necessary to prevent a race condition where a mobile has moved to a new region, but the FORWARD message has not yet propagated to the old region. It ensures that the old region will not accidentally clobber the DHT with an old value.

We have tested our implementation by moving an endpoint from one region to another, with disconnections ranging from a few seconds to a few hours. In all cases, when the mobile rejoins the network, bundles from the old region automatically are forwarded to the new region, and thence to the mobile. We are in the process of instrumenting the implementation to identify and remove performance bottlenecks.

Using the session persistence provided by OCMP [23], we have developed a simple web logging (*blogging*) application to exercise our test bed. The application runs on a wireless-enabled PDA. It polls a designated directory on the local storage of the PDA for new blog entries (simple text files or images), and sends them over the tetherless communication architecture to a proxy. At the proxy, data is extracted, and the XML RPC API to blogger.com allows the blog entry to be posted to any blog.

The protocol layering in our implementation is shown in Fig. 9. The application detects a new blog entry and passes it to the OCMP blogger plugin, which fragments the blog entry and delivers them to the TCA client running on the PDA. The client sends the bundles opportunistically through 802.11 APs to the bundle daemon on the local DTN router. This relays the bundles using DTN to the bundle daemon running on the proxy. The OCMP blogger plugin layer on the proxy registers a callback with its bundle daemon for receiving the bundles, and when they arrive, reassembles them into the original blog file. The file is delivered over the TCP/IP Internet to the blog server which publishes it. Thus, updates from the PDA become available to the world whenever the PDA is able to opportunistically connect to the Internet. Even though we are showing data movement only from the PDA to the blog, it should be clear that TCA allows comments made on the blog to be sent back to the PDA despite its moving from access point to access point, anywhere in the world.

While trivial, this application captures the essence of tetherless computing and allows us to exercise every aspect of our system. The interesting point to note is that the blog server is completely unaware of the mobility status of the receiver, and similarly, the receiver is unaware of its own mobility.

VIII. Discussion

A. Globally unique Ids and routing

Address aggregation and mobility are mutually antagonistic concepts. Address aggregation is possible only when nodes with similar addresses are topologically close, so that an address range can be assigned a common next hop; mobility means that this is precisely not the case; even if nodes with similar addresses were close by to begin with, over time they would move apart. Therefore, any scheme for mobility must support location-independent GUIDs that are either mapped to location-specific addresses, or

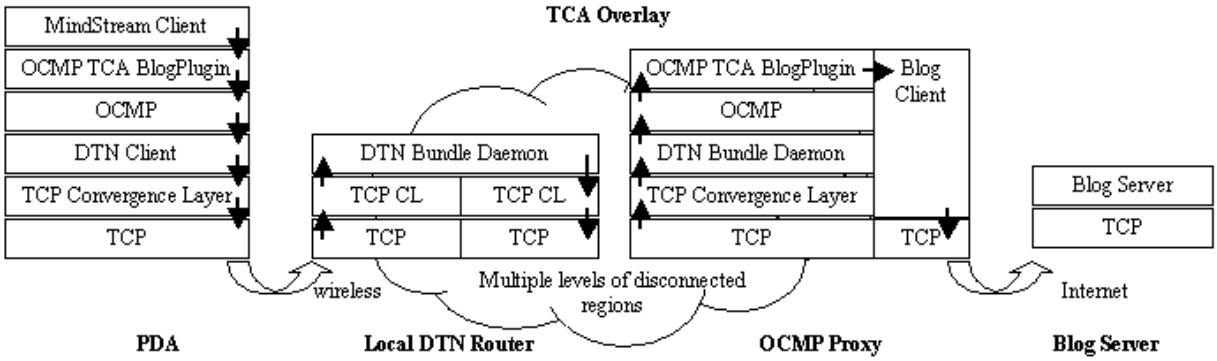

Fig. 9. MindStream blog application stack

are directly incorporated into routing tables. By the former, we mean that a node has a location-specific (aggregable) address that changes as it moves, and a lookup table maps from the GUID to the current location. By the latter, we mean that each router, for each GUID, keeps track of the next hop to reach that GUID, as in Cellular IP [5]. The latter solution is not scaleable: it would mean that the routing table size at every router would be the size of at least the number of mobiles in the system. Consequently, every mobility solution must use some form of translation from a GUID to a location-specific address, with aggregate (address-range) based routing and forwarding tables.

The current DTN architecture uses globally unique node identifiers of the form (region, admin id), where the admin id is opaque outside the region. This allows one level of aggregation, but suffers from the problem that the identifier changes as a mobile changes regions. Consequently, we need to have a globally unique unchanging identifier to identify mobile nodes, and we have introduced this into the DTN architecture.

Note that we could have mapped from a GUID directly to the location-specific address of a mobile in the Internet DHT. However, this would have required a deluge of DHT updates even for moves within a region. Our approach–to use the DHT to map from a GUID to a region–allows us to avoid DHT updates for the common case of near mobility. On the other hand, it forces us to maintain at least one more lookup table to actually determine a node's location-specific identifier. In fact, we further partition lookup in two: the VLR maps to the custodian's address, and at a custodian the LLR maps to the best next hop. This is because the only way to reach the mobile is through a custodian. So, it makes sense for the VLR to point to the custodian rather than directly to the mobile. Note that unlike the HLR, the VLR only needs to track mobiles actually in the region, so it can be much smaller.

Vanilla DTN routing ought to be enough for a custodian to choose the best next hop to a mobile's location-specific address. Since this routing has yet to be defined at the time of this writing, we use a simple GUID-based reverse path forwarding lookup table to map from a custodian to the next-hop local DTN router. Because of default routes, this table can be small.

Note also that non-mobile nodes can be assigned aggregable (location-specific) addresses. We are looking into assigning fixed nodes an aggregable address, and allocating GUIDs only for mobile nodes.

B. *Cellular telephony architectures*

Our architecture is modeled on the cellular telephony architecture. In cellular networks, multiple Base Station Systems (BSS) are grouped under a single MSC (Mobile Switching Center). A mobile's GUID is mapped in a VLR (Visitor Location Register) to an MSC. The VLR and MSC's lookup tables locate a mobile's BSS. A global HLR (Home Location Register) tracks the current location of the mobile host. TCA works similarly, with the HLR pointing to the current region, and each gateway hosting a VLR for that region.

Differences in the two architectures arise because of the partitioned network structure in TCA. Cellular telephony assumes always-available connectivity of the mobile host with the central management system. For example, although MMS (Multimedia Message Service) can be viewed as a form of delay tolerant data transfer in cellular telephony, MMS data is always stored on an MMS Router/Server (MMSRS) in the home region of the mobile. In contrast, the partitioned network in TCA requires a distributed network of custodians that are used to relay the data from one mobile host to another. Other macroscopic differences between the architectures are reviewed in Section II.A.

IX. Conclusions and Future Work

We believe that the emerging paradigm of tetherless computing requires an infrastructure that deals well with mobility and disconnection. The architecture proposed in this paper achieves the goals enumerated in Section I. Our architecture seamlessly supports mobility and disconnection even in networks where end points may never have a

direct connection to the Internet, relying instead on a disconnection tolerant overlay to ferry data to and from it. We described addressing, locationing, and routing in such networks and presented a novel technique of late-binding region names to avoid expensive lookups from disconnected nodes. We also discussed the role of GUIDs in mobility, and their impact on DTN architecture. Our proposed architecture is able to offer more than an 80-fold performance benefit for opportunistic data transfer over other conventional protocols. We now consider some avenues for future work.

Should all addresses be late bound? In some cases it might be more efficient to first do a lookup and then dispatch the data addressed directly to the current region of the receiver. Furthermore, in the case of pre-scheduled transmissions, the receiver itself can update the sender with its current location before the data transmission commences. The optimal decision on when to bind depends on factors such as the application scenarios being considered, degree of movement of the receivers, class of service, volume of data, lookup latency, cost, and resource constraints. This leads to a number of open questions: Which nodes should be looked up instead of delayed-bound? What should be the degree of caching or aggressiveness in the lookups? We have attempted to answer these questions to some extent in [10] by using service discovery protocols like Jini and SDP.

As mentioned in Section VIII, construction of a hierarchical address based routing scheme for fixed DTN nodes is an open area. We are exploring this further by looking at IPv6 to allocate addresses on hierarchical topologies for a fixed DTN network. Augmenting routing with optimization on delivery-time based metrics is another enhancement [9].

Finally, identity management is a substantial open issue in tetherless computing. We have addressed this issue using well-known techniques in identity-based cryptography in related work [22].

REFERENCES

[1] P. Barford, A. Bestavros, A. Bradley, and M. Crovella, "Changes in Web Client Access Patterns: Characteristics and Caching Implications," Boston University, Technical Report, 1998-023.

[2] K. Birman, "Building Secure and Reliable Network Applications," Manning Publications and Prentice Hall, 1996.

[3] D. Boneh and M. Franklin, "Identity Based Encryption from the Weil Pairing," Proc. Crypto 2001.

[4] C. Bovy, H. Mertodimedjo, G. Hooghimstra, and P. Mieghem, "Analysis of End-to-End Delay Measurements in the Internet," Proc. PAM 2002.

[5] A. Campbell, J. Gomez, S. Kim, and C. Wan, "Comparison of IP Micromobility Protocols," IEEE Wireless Communications Feb 2002.

[6] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, H. Weiss, "Delay Tolerant Network Architecture," Internet Draft http://www.dtnrg.org/specs/draft-irtf-dtnrg-arch-02.txt, July 2004.

[7] M. Demmer, E. Brewer, K. Fall, S. Jain, M. Ho, and R. Patra, "Implementing Delay Tolerant Networking," Intel Research, Berkeley, Technical Report, IRB-TR-04-020, Dec 2004.

[8] K. Fall, "A Delay-Tolerant Network for Challenged Internets," Proc. ACM SIGCOMM 2003.

[9] S. Jain, K. Fall, and R. Patra, "Routing in a Delay Tolerant Network," Proc ACM SIGCOMM 2004.

[10] S. Keshav, P. Darragh, A. Seth, and S. Fung, "Protocols for Tetherless Computing," University of Waterloo, Technical Report, 2005.

[11] J. Kurose and K. Ross, "Computer Networking," Addison Wesley, 3rd Edition, pp.271, 2004.

[12] Y. Mao, B. Knutsson, H. Lu, and J. Smith, "DHARMA: Distributed Home Agent for Robust Mobile Access," Proc. IEEE INFOCOM 2005.

[13] A. Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: An Approach to Universal Topology Generation," Proc. IEEE Mascots 2001.

[14] A. Mishra, M. Shin, and W. Arbaugh "Pro-active Key Distribution using Neighbor Graphs," IEEE Wireless Communications, Feb 2004.

[15] R. Moskowitz, P. Nikander. P. Jokela, and T. Henderson, "Host Identity Protocol," http://www.potaroo.net/ietf/ids/draft-ietf-hip-base-00.txt, 2004.

[16] J. Ott and D. Kutscher, "Drive-Thru Internet: IEEE 802.11b for Automobile Users," Proc. IEEE INFOCOM 2004.

[17] J. Ott and D. Kutscher, "A Disconnection-Tolerant Transport for Drive-thru Internet Environments," Proc. IEEE INFOCOM 2005.

[18] A. Pentland, R. Fletcher, and A. Hasson, "Daknet: Rethinking Connectivity in Developing Nations," IEEE Computer, 37(1):78-83, 2004.

[19] C. Perkins, "IP Mobility Suport for Ipv4," http://www.ietf.org/rfc/rfc3344.txt, Aug 2002.

[20] L. Peterson, T. Anderson, D. Culler, and T. Roscoe, "A Blueprint for Introducing Disruptive Technology into the Internet," Proc. ACM Hotnets 2002.

[21] S. Rhea, B. Godfrey, B. Karp, J. Kubiatowicz, S. Ratnasamy, S. Shenker, I. Stoica, and H. Yu, "OpenDHT: A Public DHT Service and Its Uses," Proc. ACM SIGCOMM 2005.

[22] A. Seth and S. Keshav, "Practical Security for Disconnected Nodes," Submitted for publication.

[23] A. Seth, S. Keshav, and S. Bhattachharaya, "Opportunistic Data Transfer Over Heterogeneous Wireless Access Networks," Manuscript, Work in progress.

[24] R. Shah, S. Roy, S. Jain, and W. Brunette, "Data MULEs: Modeling a Three-Tier Architecture for Sparse Sensor Network," Proc. IEEE SNPA 2003.

[25] A. Snoeren and H. Balakrishnan, "An End-to-End Approach to Host Mobility," Proc. ACM MOBICOM 2000.

[26] H. Soliman, C. Catelluccia, K. Malki, L. Bellier, "Hierarchical Mobile IPv6 mobility management (HMIPv6)," http://www.ietf.org/internet-drafts/draft-ietf-mipshop-hmipv6-04.txt, 2004.

[27] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana, "Internet Indirection Infrastructure," Proc. ACM SIGCOMM 2002.

[28] V. Zandy, and B. Miller, "Reliable Network Connections," Proc. ACM MOBICOM 2002.

[29] W. Zhao, M. Ammar, and E. Zegura, "A Message Ferrying Approach for Data Delivery in Sparse Mobile Ad Hoc Networks," Proc. ACM MOBIHOC 2004.

[30] "MAPLE 6.1," Maplesoft www.maplesoft.com.