Open access • Proceedings Article • DOI:10.1145/1162654.1162659

Integrating DTN and MANET routing — Source link

Jörg Ott, Dirk Kutscher, Christoph Dwertmann

Institutions: Helsinki University of Technology, University of Bremen

Published on: 11 Sep 2006 - Workshop Challenged Networks

Topics: Dynamic Source Routing, Destination-Sequenced Distance Vector routing, Link-state routing protocol, Zone Routing Protocol and Optimized Link State Routing Protocol

Integrating DTN and MANET Routing

Jörg Ott
Helsinki University of Technology
Networking Laboratory
jo@netlab.tkk.fi

Dirk Kutscher, Christoph Dwertmann
Universität Bremen
Technologiezentrum Informatik
{dku|nermal}@tzi.org

ABSTRACT

Mobile Ad-hoc Network (MANET) routing protocols aim at establishing end-to-end paths between communicating nodes and thus support end-to-end semantics of existing transports and applications. In contrast, DTN-based communication schemes imply asynchronous communication (and thus often require new applications) but achieve better reachability, particularly in sparsely populated environments. In this paper, we suggest a hybrid scheme that combines AODV and DTN-based routing and allows keeping the AODV advantage of maintaining end-to-end semantics whenever possible while, at the same time, also offering DTN-based communication options whenever available—leaving the choice to the application. We present our protocol and system design, particularly including the interaction of AODV and DTN, demonstrate achievable performance gains based upon measurements, and report on initial experiments with our implementation in an emulation environment.

Keywords

Delay-tolerant Networking, Mobile Ad-hoc Networking, Routing

1. INTRODUCTION

Mobile ad-hoc networks may complement infrastructure-based wireless networks and allow mobile and nomadic users to obtain access to Internet services or to interact directly with one another even when they are outside the coverage area of cellular networks or wireless LAN hotspots. Similarly, MANETs may enable communication between vehicles, sensors, and other mobile equipment without the need to deploy an infrastructure network. Numerous dedicated routing protocols have been and are being developed to establish and maintain reachability between communicating nodes in such dynamic environments. Nevertheless, the practical use of MANETs is mostly envisioned in closed (sensor) networks and other controlled (e.g., tactical) deployments but is not widespread elsewhere. While market forces are probably the most important factor, we can also identify two technical inhibitors:

(1) Running Internet protocols between mobile peers requires an end-to-end path to be found and sustained for a sufficient period of time to allow the respective application interactions to complete. In MANETs, the existence of such a path obviously depends on mobile node density and movement dynamics. Particularly personal communication devices (PDAs, cellphones, etc.) will be heterogeneous (with potentially incompatible feature sets) leading to reduced effective mobile node density, to which individual settings and unwillingness to cooperate may add. All this makes communication across several hops unreliable. (2) Furthermore, running the most dominant transport protocol (TCP) over a series of wireless (ad-hoc) links (e.g., in wireless meshes or MANETs) will yield poor performance with an increasing number of wireless hops [20].

Both aspects can be addressed by using ad-hoc networking approaches following the concept of delay-tolerant networking (DTN) [5]. In fact, numerous proposals have been developed that use asynchronous hop-by-hop communications to avoid the need for an end-to-end path (1) and thereby (implicitly) also address the issue of (2). However, these usually force the applications into the DTN-inherent asynchronous communication paradigm—and often require specific application protocols designed for this purpose.

For many Internet applications, we have to acknowledge that DTN-based asynchronous messaging, even if feasible, may only be second choice and that users would prefer end-to-end communications: The latter are likely more interactive and may exhibit better performance, deliver immediate results, or applications simply depend on end-to-end transport connections (as with TLS-based security mechanisms). Nevertheless, depending on the particular situation, an application and ultimately its user may prefer asynchronous communications over not communicating at all.

In this paper, we propose a hybrid approach to IP-based communications in mobile ad-hoc networks: we combine IP layer MANET routing using AODV [19] and DTN-based forwarding according to the DTNRG bundle specification [21]. We use AODV to determine the (non)availability of an end-to-end path between two peers and, in parallel, learn about available DTN routers in the surrounding environment. We provide the available communication options to the application which may take a simple decision (end-to-end path available or not) or weigh different options based upon their parameters following a more sophisticated reasoning. The application chooses between end-to-end and DTN-based communication and may even provide routing hints for the latter. Our scheme accommodates device heterogeneity by not requiring all mobile nodes except for the involved endpoints to be DTN capable and thus allows ad-hoc paths across non-DTN-capable nodes to help bridging the otherwise existing gaps between DTN nodes. This enhances DTN reachability while preserving the option of end-to-end communications and thus increases the overall communication capabilities.

In section 2, we briefly review related work from mobile ad-hoc and delay-tolerant networking and present two application scenarios in section 3 motivating the integration of the two communica-

SIGCOMM'06 Workshops September 11-15, 2006, Pisa, Italy.

tion schemes. We describe our hybrid routing approach in detail in section 4. We introduce a test setup and our implementation and report on initial performance findings in section 5. We conclude this paper with a summary and suggestions for future work in section 6.

2. BACKGROUND AND RELATED WORK

2.1 Mobile Ad-hoc Networking and AODV

Research on mobile ad-hoc networking has led to the development of a variety of routing protocols and continues to improve these. MANET routing protocols can be roughly divided into proactive (such as DSDV [18] and OLSR [1]), reactive (such as AODV [19]), hierarchical (such as ZRP [6]), and geographic routing [24]. Since our motivation is to improve communications in sparse networks that will be highly dynamic, proactive routing protocols appear to be unsuitable and structuring the few reachable but dynamic nodes into a hierarchy does not appear necessary either. We also do not assume geographic location information to be available. As a result, we have chosen AODV as a reactive routing protocol, the semantics of which also matches our on-demand discovery of close-by DTN nodes. AODV is well-documented as experimental RFC 3561 and provides the necessary hooks for our extensions.

AODV [19] combines on-demand route discovery, route maintenance, and caching to determine paths from a source to a destination. AODV uses three messages types: a *Route Request* (RREQ) is broadcast by a source to discover a route to a particular destination—using an expanding ring search (by increasing IP TTL) to limit the network traffic incurred. Each node forwarding an RREQ creates (bidirectional) local forwarding state for return routing of a later response back to the originator. If an RREQ reaches the destination (or another node that already knows a valid path to the destination), the respective node responds with a *Route Reply* (RREP) which is forwarded back to the originator. A passing RREP packet activates the previously created routing entries along the path (which are timed out otherwise) so that IP packets can flow. An RREQ that exceeds its lifetime is silently discarded; route discovery failures are detected by a timeout mechanism.

The route discovery may implicitly establish a reverse route end-to-end. Each routing entry is associated with a lifetime that is refreshed for each packet that passes. If routes break because links or nodes disappear (noticed, e.g., from missing Hello-style L2 or AODV messages or from link layer indications that a data packet cannot be delivered), a local repair mechanism can be invoked. If a route cannot be maintained, *Route Error* (RERR) messages are sent along the affected paths to invalidate the cached routing state.

2.2 Delay-tolerant Networking

While the "classical" MANET protocols seek to establish an end-to-end path between the communicating peers and thus enable higher layer end-to-end protocols to operate largely unchanged, the need for operation in mobile ad-hoc networks with sparse node distribution has led to the development of specific application protocols that work using asynchronous communications and do not require an end-to-end path at any point in time. Different approaches have been studied for sensor networks [10, 25, 23], interpersonal communication [8], and Internet access in remote areas [3, 17] but also in support of mobile Internet access [13, 15], among others.

In this paper, we focus on the DTN architecture for asynchronous communications as developed in the DTN Research Group (DTNRG) of the Internet Research Task Force (IRTF). Rather than exchanging (small) packets end-to-end, DTN endpoints use messages of arbitrary size (*bundles*) forwarded by DTN routers hop-by-hop from the source to the destination. The DTN *bundle protocol* operates above the transport layer and may interconnect different internets with arbitrary underlying protocol stacks [5]. Status reports may convey information about the delivery progress of a bundle from intermediate routers as well as receipts from the receiver to achieve end-to-end semantics [21]. DTN endpoints and applications are identified by *endpoint identifiers (EIDs)* specified in a URI-style format: *scheme:specific-address*. The *scheme* defines the scope within which the *specific address* is interpreted.

Routing in DTNs has been studied in general [9] as well as in the context of specific mobile ad-hoc networking environments. For the latter, DTN routing often relies on information replication for forwarding bundles to maximize delivery probability and/or minimize transit time. Such forwarding schemes follow, for example, the concept of *epidemic routing* [22]. Numerous variations and improvements have been developed such as [12, 11, 7].

While the issue of node mobility and non-available end-to-end paths have been recognized for MANET environments and DTN routing schemes for mobile networks are being studied and improved, the present approaches have in common that they take either a pure end-to-end packet-based approach to communications or use exclusively asynchronous hop-by-hop information exchange. We believe that schemes coupling DTN concepts and opportunities for end-to-end communications may help applications and may also benefit DTN routing itself as we outline in the next section.

3. APPLICATION SCENARIOS

As noted above, many of today's IP-based applications assume some notion of end-to-end operation in their protocol, even if they explicitly support intermediaries. For example, when sending mail via their outbound SMTP server, mail clients do so synchronously and learn about preliminary acceptance or rejection of the mail. When sending an instant message to another party, an end-to-end transport to the recipient allows indicating that the message got delivered while an end-to-end interaction with a messaging server at least confirms that, similar to the mail server, another (typically well-connected) entity will take care of further delivery. When accessing a web page via a proxy, the request-response interactions are carried out end-to-end between the client and the origin server if needed to immediately determine availability and freshness of the requested resource. All three applications could, in principle, also operate using exclusively asynchronous DTN operations. Obviously, the applications would need to either be enhanced to support DTN or would require intermediaries (e.g., SMTP or HTTP proxies/gateways) to perform the necessary adaptation locally as we have discussed for HTTP elsewhere [15, 16]. In either case, we assume that both involved endpoints are DTN-capable.

Even a DTN-capable application may prefer end-to-end communication whenever possible: for better performance, for preserving end-to-end semantics with immediate responses, or simply because some interactions cannot be performed using DTN. But asynchronous DTN-based communication may be considered a suitable backup in many cases [14]. Applications should therefore be notified about the available communication options which can be determined when attempting to obtain a route to the target (via AODV). Such notifications should comprise possible paths for end-to-end communications (plus their hop counts) and information about intermediate DTN nodes (including hop counts and DTN routing metrics) so that the application can make a conscious choice (autonomously, based upon user policy, or after inquiring the user). End-to-end communication (if available) may be chosen if the number of wireless hops and the expected route stability are deemed suitable for the size of the message to be transmitted. Otherwise, the application may opt for hop-by-hop communica-

tions (and provide hints about possible next hops to the DTN subsystem) or decide that the intended transaction is not suitable for asynchronous interaction, report failure, and retry later.

The same applies to DTN routing (considering a DTN router such as *dtnd* [4] as an application). Like an application trying to reach its peer, a DTN router attempts to find and contact peers for bundle forwarding. For this purpose, the DTN router may rely on link-specific adaptation layers or it may use, e.g., the TCP convergence layer [2]. While the former may limit interactions to directly adjacent nodes, the latter allows reaching a peer across an arbitrary number of IP hops. However, if a known peer is many wireless hops away, performance degradation and route instability may make direct ("end-to-end") communication infeasible. The hop count to the target peer (or its present non-reachability) can be learned from the AODV route search. This process may also yield closer available DTN routers (of which those more than one hop away would not be found by link layer interactions only) which can then be used by the DTN router for its forwarding decision.

Both classes of applications can be supported by minor extensions to AODV (and possibly other ad-hoc routing protocols) and a minimal integration with DTN routing, in the DTN router and/or the application, as we present in the next section.

4. INTEGRATING DTN AND AODV

DTN and AODV fulfill complementary and independent functions and operate at different layers: AODV dynamically determines and maintains an end-to-end path between two peers wanting two communicate in MANETs while DTN is typically used when the existence of such a path is rather unlikely so that using asynchronous communication is advisable. When running DTN on top of IP, DTN communication exploits the multihop end-to-end routing between two DTN nodes. However, as discussed above, DTN nodes do not depend on underlying multihop communication capabilities but may also perform the respective routing functions hop-by-hop at the link layer. Nevertheless, the availability of multihop routing from lower layers appears beneficial because it essentially enhances the view of the network for DTN nodes and thus their potential reach—which is likely to increase the number and diversity of possible next DTN hops for bundle forwarding.

This is depicted in figure 1 where the inner dotted circle marks the immediate neighbors of DTN node O reachable via plain L2-based DTN forwarding while the outer dashed circle shows the reach if up to three hops are supported by mobile ad-hoc routing. At the same time, including DTN-based routing expands the reach of the originator via multiple DTN hops to the target T which would not be reachable with AODV alone.

In this section, we present our approach towards a minimal integration of DTN and AODV that achieves the three goals sketched above: 1) to extend the view of DTN nodes and thus allow for greater flexibility in routing decisions, 2) to enable communicating asynchronously with application peers otherwise out of reach, and 3) to allow applications—but also DTN routers—to dynamically trade off immediate interaction for reachability.

4.1 Conceptual Overview

We assume that an originating node O wants to communicate with a target node T as depicted in figure 1. Ideally, such an interaction should be carried out end-to-end between O and T without involving intermediate nodes with more than plain IP routing.

As noted in section 2, AODV will initiate route discovery on-demand when a packet needs to be transmitted to a target IP node and no valid cached route is available. During the route discovery process, an AODV node performs an expanding ring search for the target node in its surrounding territory during which RREQs are processed and forwarded by all directly and indirectly neighboring nodes up to a maximum diameter or until the target node is found.

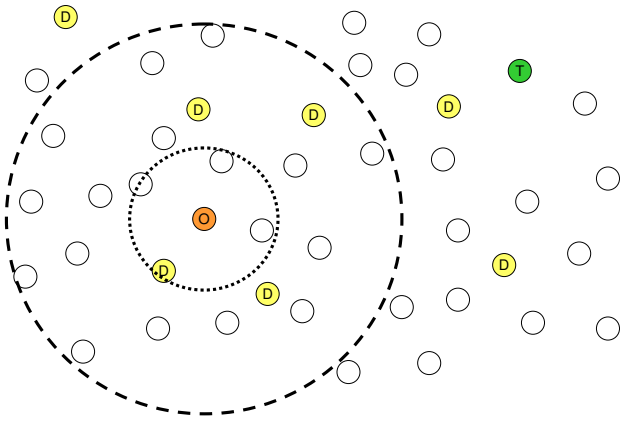

Figure 1: Increasing the Reach of Communications

We augment this route search process to have each AODV node on the path note—in the request and later in the reply—whether it supports DTN routing and, optionally, report its DTN routing metrics for the target (and originator) node. Since this is part of the AODV route search, no additional messages are needed; also, caching and route invalidation can work for DTN-related routing information as they do for AODV paths. This essentially augments AODV route discovery to become an implicit service discovery mechanism for DTN routers. DTN-specific timeout handling ensures that route replies containing DTN routers are returned even if the final target cannot be reached. Caching and timeout mechanisms for the specific DTN extensions prevent the network from being flooded with repeated DTN route information.

When the originator receives the results of the route discovery, it may or may not find AODV routes to the target as well as zero, one, or more available DTN routers in reach. As discussed in section 3, the originator then decides whether to use end-to-end IP communication, to go for hop-by-hop DTN messaging, or to declare failure for this point in time (with the option of retrying later as deemed acceptable by the application). If DTN is used, the originator evaluates the obtained DTN routing information from the route discovery to determine a (set of) suitable next hop(s). Note that we assume that a DTN routing protocol is in place that is capable of taking the appropriate forwarding decisions based on the reachability information supplied by our AODV extensions.

4.2 Extensions to AODV

Three types of processing extensions are required to support DTN route(r) discovery in AODV: the route discovery needs to be expanded (the header fields as well as generation and processing of RREQs and RREPs); an additional timeout mechanism is required to deal with unsuccessful route requests; and specific error handling needs to be introduced to deal with route failures. Each AODV node has to maintain additional state information per AODV route during the search phase and after its establishment.

Figure 2 depicts the two proposed extension headers[1] for AODV to carry the DTN-specific information: a) The *DTN router info*

[1]The extension header type numbers are chosen at random from the non-mandatory part of the available numbering space.

header extensions is appended to both RREQ and RREP packets by AODV routers along the path to indicate their DTN router capability. It include a node's contact information (IP address and port number), its distance from the originating node (hop count), and a timeout handling flag T (discussed below). b) The DTN EID extension may be used by the originating endpoint to specify the DTN layer source and target EIDs so that a reference for providing detailed routing information is available. If DTN EID extensions are present, the DTN router info extension may also include optional routing metrics (dark shaded fields): the D bit indicates whether the routing information refers to the source (D=0) or the target (D=1) EID, the remaining fields allow a router to specify its EID and variable length routing metrics for one or more routing protocols.[2] Since the basic processing rules for AODV extensions are independent of whether or not routing metrics are included, the remainder of this subsection does not consider these further.

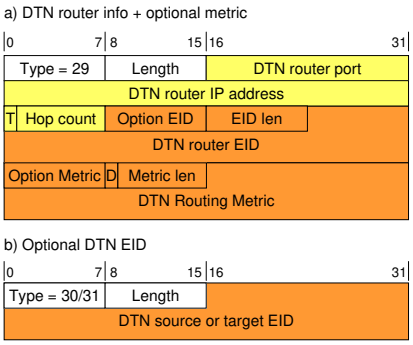

Figure 2: AODV Packet Extensions

4.2.1 RREQ Generation and Processing

When generating an RREQ packet during route discovery, the originating node includes a DTN Router Info extension with its own contact information and the T bit set to 0 and then broadcasts the message to its neighbors. An AODV node not supporting the DTN extensions will simply forward the RREQ packet after local processing of the fields it understands. A DTN-capable AODV node adds another DTN Router Info extension including its contact information and its hop count to the originator (up to MTU size). The T bit is set to 1 unless another DTN Router Info has it already set. The node also copies the list of DTN routers into its local entry created for this route search and then forwards the modified RREQ packet. This processing results in the RREQ packets accumulating information about all available DTN routers along a given path.

4.2.2 RREP Generation and Processing

By the target: As soon as the target is reached, a corresponding RREP message is generated. If the final destination is DTN-capable (which it usually is), it copies the DTN Router Info fields (and the DTN EID fields, if present) from the RREQ into the RREP packet, adds its own contact information, and sends the RREP to the predecessor on the path. If the final destination is not DTN-capable (or does not support the proposed DTN-AODV integration), it will simply generate a plain RREP without the extension headers.

As the RREP packet is sent back along the path from the target towards the originator, all AODV nodes update their AODV routing information accordingly. The DTN-capable nodes additionally extract DTN router information for their DTN routing tables. If a DTN-capable node observes from its local state that an RREP does not contain any DTN Router Info extensions even though it added such information to the corresponding RREQ it knows that the target is not AODV-DTN-capable, recreates the previously included DTN Router Info fields, and adds them to the RREP message. This will ensure that information about all DTN routers on the path is returned to the originator.

By a node N having a route to the target: If a node N is reached that knows a valid path to the destination, it will usually terminate the route search and send an RREP. This is what happens if node N is not DTN-capable and the resulting RREP will not contain any DTN Route Info extensions so that the procedure defined in the previous paragraph will be executed. On one hand, this implies the limitation that only part of the path may be searched for DTN routers; on the other hand, an end-to-end route is most likely to exist that can be used for end-to-end or DTN-based forwarding and thus likely outweighs this "deficiency".

If node N is DTN-capable, it consults its routing entry for the destination and determines the remaining number of hops to go as well as DTN routing information cached from a previous route discovery. If such DTN information is available, it is appended to the DTN Router Info fields from the RREQ and the complete set is included in the RREP message which is returned right away. If no DTN information is available (e.g., because a previous route discovery did not involve a search for DTN nodes), the node may decide either to copy the DTN Router Info fields from the RREQ and immediately generate an RREP or to forward the RREQ after local processing along the existing route towards the target. One decision criterion may be the remaining path length towards the target: if this is only a few hops seeking further DTN intermediaries may not be worthwhile but this may be different for longer paths.

4.2.3 Route Discovery Failure

If the target node is not found by AODV for a given iteration of the expanding ring search, no RREP messages are generated but RREQs are silently dropped. This means that no response would be generated at all and that no DTN nodes can be returned. Therefore, DTN-capable nodes follow a slightly modified procedure for RREQ packets that contain DTN Route Info extensions. For each round of the expanding ring search, one DTN-capable router along each broadcast search path (if any are in reach) is determined to watch over the route discovery process. This router starts a timer[3] when the RREQ packet passes. If the responsible router observes an RREP it cancels the timer. If the timer expires (i.e., no route was found), the router generates a "DTN-only" RREP which indicates a lifetime of zero for the indicated AODV route[4] but includes the DTN Router Info and EID fields from the router's local state.

To determine the responsible router, a DTN-capable router uses the T bit to indicate whether it considers itself to be responsible. To avoid avalanches of DTN-motivated RREP packets, only the first DTN router on the path does so—subsequent ones observe that a predecessor has already set this bit in its entry and do not perform this extra function. In order to allow more than just the nearest router to be to detected, each router only performs this function

[2]For example, reporting the routing information in PRoPHET[11] for a single target with an EID of 32 bytes length requires some 72 bytes when using PRoPHET's regular routing information exchange protocol. Note that this can be optimized for AODV piggybacking as, e.g., the actual routing metric is only two bytes in size and the EIDs are already included in the message.

[3]$NetTraversalTime - 2 * HopCount * NodeTraversalTime$

[4]Which we assume to be forwarded without route state creation by non-extended AODV nodes.

once during a particular route discovery. That is, a router that was responsible router for one round of the expanding ring search, will not become one for all subsequent rounds (which it indicates by setting the T bit in its entry to 0). While this limits discovery to the first n DTN-capable routers in an expanding ring search with n iterations, it also constrains amount of reply traffic back to the originator (which we consider more important since most paths of the route discovery will usually not succeed).[5]

4.2.4 Route Collection at the Sender

From the RREP packets returned from the above steps, the originator gathers (1) possibly one or more routes to the target node and their respective path lengths and (2) in addition a collection of zero or more reachable DTN-capable routers (a) along the path to the destination and (b) within its environment (depending on the number of iterations and the lifetimes of the expanding ring search[6]).

If no RREP packet is received within the path discovery time, no communication options exist and the originator's attempt has failed. If an end-to-end route exists, the originator can use end-to-end communications with the target. If also one or more DTN routers are found, the originator has the option to use DTN instead of end-to-end communication. When doing so, it would preferably choose DTN routers on intermediary nodes along the path to the destination. Finally, if no end-to-end path is available but one or more DTN routers are found, the originator may choose to either use or more of these for asynchronous communication or declare failure. Which information is available and how it may be used is discussed in the next subsection.

4.3 Interaction of DTN and AODV

With the above scheme, AODV is used as a vehicle to locate DTN routers as possible optimization for communication or fallback in case no end-to-end path can be determined. In the following, we consider two cases that essentially differ in how much information from DTN routing is available to and communicated via AODV and how much the DTN routing algorithms can exploit the knowledge about underlying AODV routes. In both cases, we strive for a minimal integration across the two layers of DTN-based and IP-based routing to minimize mutual dependencies and to allow replacing AODV in the future by other ad-hoc protocols (for which similar extensions can be defined).[7]

4.3.1 Simple Case: Locating DTN Routers

In the simple case, DTN router discovery is limited to determining available DTN routers in the vicinity of the originator node. On a transit node, availability is limited to a "yes/no" decision and can be statically configured on each intermediate node: if DTN routing is supported, the node will add itself to the list of available DTN routers when an RREQ or RREP passes by as described above. Apart from this, AODV operates independently of DTN routing. On the endpoints, the resulting information about nearby DTN routers is gathered and the additionally available peers are fed into the local DTN router. For taking a routing decision, e.g., at the originator, an independent procedure is required that, depending on the routing algorithm, may take into account past information about the DTN nodes, may explicitly contact them for exchange of reachability information, and/or may simply decide to follow some epidemic routing protocol and disseminate the message to a subset or all of these nodes. One important criterion for forwarding may be how many wireless hops the respective DTN routers are away.

4.3.2 Advanced Case: Supporting DTN Routing Info

In a more advanced case, the discovery phase is also used to obtain routing information specific to the DTN routing protocol. To achieve this, the RREQ packets include the target EID for the DTN message. Upon receipt of such a message, a DTN-capable transit node consults its internal DTN routing table and determines the routing-protocol-specific metric for the target EID which it adds both to the RREQ and later to the RREP packets in addition to its reachability information. The originator may use the information returned in the RREP to update its local routing tables and decide which next DTN hop(s) to choose for forwarding the message. The target (or the next DTN hop chosen by the originator) has also learned any further DTN hops along the way from the RREQ message and may cache this information.[8]

4.3.3 Implications for DTN Routing

While AODV is used for layer 3 routing, different routing protocols are used at the DTN layer, including those mentioned in section 2.2. The role of AODV in DTN routing is limited to the two aspects described above: 1) aiding in discovering potential DTN peers reachable at a given point in time for forwarding bundles and 2) possibly providing further hints to choose a subset of these for further consideration in a particular forwarding decision.

A DTN router obtaining a number of peer routers via 1) still has to determine for which targets they are suitable next hops. This is the task of a DTN routing algorithm and the DTN router may need to contact these peers and exchange routing information defined for the respective routing protocol. It is only after this exchange—which may also be used to determine whether a peer is acceptable from a security perspective—that the originator can take a forwarding decision for a bundle. In this sense AODV is, as described above, simply a means to extend the reach of a DTN router.

If the advanced mechanism is used, 2) may offer important additional information for the DTN router: Since the AODV route search may yield a large number of DTN nodes, it may be too costly (in terms of bandwidth and incurred delay) to engage into an individual routing exchange with each of these. DTN routing metrics returned by the AODV may thus aid in choosing a particular subset of DTN peers to contact further. The information gained may be useful even beyond the next hop, particularly if an end-to-end path has been discovered but was deemed too long. In this case, the next hop DTN router can be chosen from the ones on the path to the target (which may be reflected accordingly in the local routing table) and, assuming a suitable "source routing" mechanism in DTN, the originator could also provide hints for further routing steps to the next hop DTN router when transmitting the bundle.

[5] Explicit timer randomization coupled with reply damping mechanisms require further study.

[6] RFC 3561 suggests four iterations for IP TTLs 1, 3, 5, and 7.

[7] Note that similar considerations could be applied to infrastructure-based networks and thus to IP routing in general—but there does not seem to be an immediate need for dynamic DTN support in rather stable IP networks, except maybe for wireless access.

[8] As a further optimization, the originator may also supply its own in addition to the target EID in the RREQ packet so that DTN routers along the path can also add reverse path routing metrics for the originator—which would allow the target to update its DTN routing tables, too. The extension format supports this by a single bit to indicate whether a routing metric applies to the originator or target EID. Furthermore, each DTN router may optionally add its own EID and, in the future, capability indicators (e.g., *custody transfer* support) to provide further information for a DTN routing protocol. However, any such extension comes at the expense of increasing the size of the exchanged packets so that the MTU size limit may be reached earlier and error probability may grow; such trade-offs require further investigation.

4.4 Application Interface

The above integration of DTN and AODV routing is capable of collecting information about different options to reach a target but it is the application on the originator that must decide how to use this information. Therefore, the application needs to be able to inquire and obtain the extended routing information from the lower layers and also be able to feed part of this information, e.g., routing hints, to (not just) its local DTN router. A DTN router may itself be one application that makes use of the AODV routing to reach its peers and may invoke AODV mechanisms to obtain routes to or hints about its peers. Both scenarios are shown in figure 3.

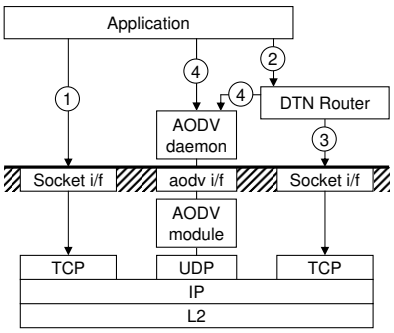

Figure 3: Application Interfaces for DTN-AODV Integration

As can be seen from figure 3, the AODV module comprises a kernel module for low level interactions with other kernel structures and for packet handling as well as a user-level daemon that implements the protocol logic of AODV and maintains all the complex data structures. AODV operates independently of and invisibly to the applications: if an application sends an IP packet to a target (e.g., a TCP SYN as a result of a connect() system call) for which no route exists, the AODV module becomes active and tries to find one—which results in the respective system call to succeed or to fail. The standard socket interface (1, 3 in the figure) does not provide any further details about the failure and the route results obtained from AODV cannot be connected easily to the respective socket descriptor. Therefore, we suggest to tap into the AODV information available in the user daemon using a simple asynchronous IPC interface to supply and retrieve DTN-related information (4). This approach also allows to continuously monitor changes to the AODV routing information, for example, to incrementally retrieve the results returned during a route search.

After completion of the initiating system call, the respective application can retrieve the DTN-related information and AODV parameters by accessing the corresponding target node entry and decide whether to use end-to-end connectivity (if available) or asynchronous DTN communication. If the application is the DTN router, all necessary data are available and the forwarding process can be initiated.[9] If the application is disjoint from the DTN router, its interface to the latter (2) should provide the capability to provide routing hints based upon the knowledge just gained from the AODV route discovery because the DTN router may not be able to correlate the target DTN EID with information from the AODV database (if it has access to this information at all). Adding a variant of *(loose) source routing* may provide a mechanism for an application to make routing suggestions for the first and also subsequent hops to the DTN router based upon application-specific knowledge.

[9]Possibly after updating its local routing tables which may involve contacting the intended next hop(s) as noted above.

Open issues include how to efficiently associate DTN parameters (e.g., the source and target EIDs) with an AODV route search; how to initiate AODV route discoveries to a target node without invoking the transmission of an IP packet; how to best obtain asynchronous notifications about (incremental) changes to AODV state in the context of route searches; and how to create a single API for the application to obtain and provide DTN-related information.

5. PERFORMANCE OBSERVATIONS

We are investigating our proposed approach in two ways: we have validated the (implicit) assumption that message-based hop-by-hop forwarding above the transport layer will improve the application goodput in settings with multiple wireless hops compared to end-to-end TCP and we are running a combined implementation of DTN and AODV in an emulation environment for functional and (so far only minimal) performance assessment.

5.1 Multihop Wireless Measurements

We have created a static measurements setup as depicted in figure 4 consisting of 6 Linksys WRT54GS access points running OpenWRT Linux (White Russian RC4), an IBM T41p laptop running MS Windows XP SP2 and Cygwin and using its internal 802.11 card, and a 64-bit Intel PC running SuSE Linux 2.6.8-24.19, all interconnected by a switched 100 Mbit/s-Ethernet-based VLAN for management purposes. All wireless equipment was set to operate in ad-hoc mode using IEEE 802.11b/g.[10] We have used our tool *tcpx* for traffic generation (the client running on the laptop and the server on the PC) to create traffic at maximum rate in either direction by sending 1460 byte TCP segments. By means of routing re-configuration we selected any number of wireless hops (1, ..., 6) in both directions between client and server to measure the achieved net application data between the two *tcpx* instances for 60 s. The results roughly confirm earlier findings [20]. For client to server communication, the mean data rate decreases significantly from 7.4 Mbit/s (1 hop), 3.4 Mbit/s (2), 2.9 Mbit/s (3), 2.0 Mbit/s (4) down to 1.3 Mbit/s (5) and 1.4 Mbit/s (6 hops) with up to 30% variation (five to ten measurements each).

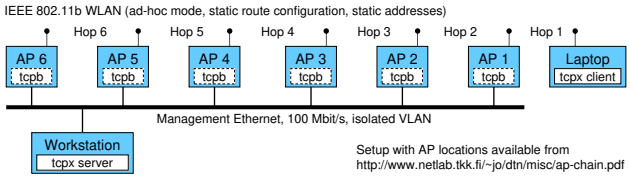

Figure 4: WLAN AP Measurement Chain

To determine the impact of DTN routing nodes, we have set up intermediaries on some or all of the access points using our simple TCP bridge *tcpb* that accepts and forwards connection requests and subsequently exchanged data. As reference, we have used TCP connection splitting where received segments are immediately forwarded to the next hop. To imitate bundle communications, we have then configured *tcpx* to generate bundles of data (10, 20, and

[10]Other access points on the campus overlapped in frequency range and we could not control WLAN-incurred traffic and other background noise when conducting our measurements. We therefore chose a relatively quiet period in June 2006 and use the results only as rough indicator rather than for detailed quantitative analysis. (An earlier experiment carried out during "busy" April 2006 yielded similar observations but achieved only some 50% of the data rate and much higher (more than 50%) variation.)

40 TCP segments of 1460 bytes each) in regular intervals ranging from 25 ms to 400 ms and *tcpb* to first fully receive a complete bundle and then forward it. We performed a total of 700+ measurements. Figure 5 summarizes some results for the above setup with six wireless hops (mean values based on three measurements each): A positive impact of simple TCP connection splitting is observable in most settings but the performance gain over end-to-end TCP is limited. For bundle-based transmission, the results consistently show a significant performance increase in virtually all cases, roughly doubling with three or more intermediaries at a targeted offered load of 4.7 Mbit/s[11] compared to end-to-end TCP. The plotted bundle transmission curves are representative for all our measurements when the transmission capacity gets saturated. A clear correlation to the bundle size alone could not be observed.

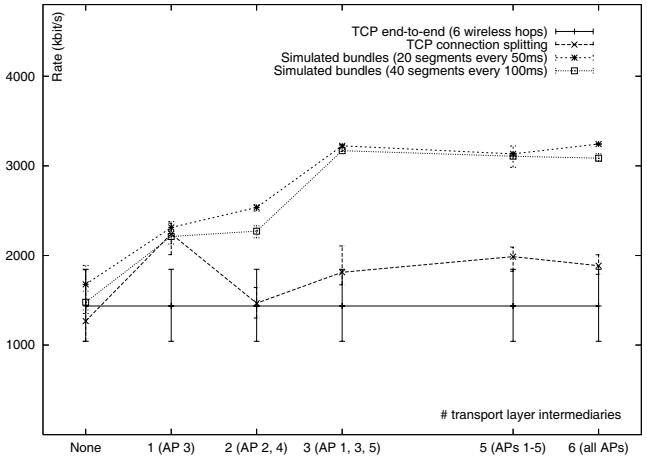

Figure 5: Measurement Results for 6 Wireless Hops

The above measurements have neglected per DTN router processing (which should just add a constant delay). To obtain performance values using a real DTN implementation, we have used the *dtnd* ported to the Linksys access point by ENST[12] instead of *tcpb* in the same setup as above. Using the same bundle sizes and frequencies (with *tcpx* for traffic generation/reception, *dtntcp* [15] for conversion to/from bundles, and local *dtnd*s running on the laptop acting as sender and receiver), we observed only 1 Mbit/s $\pm 20\%$ irrespective of the number of *dtnd* intermediaries. Further investigations revealed that the performance of the (non-optimized) *dtnd* on the Linksys APs appears to be the bottleneck so that a direct comparison to the above measurements is not meaningful.

In summary, we note that bundle-based hop-by-hop communication can improve performance notably compared to end-to-end TCP and plain connection splitting—so that the basic approach of introducing DTN routing into a multihop wireless network may provide a clear throughput gain in addition to the expected improvement in reachability and thus may be an interesting alternative for applications even if an end-to-end path exists.

5.2 DTN-AODV Emulation

We have implemented a sample application for a hybrid AODV-DTN network for validating our approach. The main objective is to demonstrate the general benefits of enhancing an ad-hoc-network-based infrastructure by adding DTN transport services, with respect to robustness and performance. The key functions are determining end-to-end connectivity, dynamically switching to DTN transport when no end-to-end connectivity is available, dynamically locating suitable DTN-enabled nodes, and dynamically adapting local DTN routing tables to the changing network topology.

Our system comprises five main components: 1) the AODV routing subsystem[13] (enhanced by our DTN extensions); 2) DTN bundle routers (*dtnd* [4]); 3) our DTN transport layer proxy for TCP and HTTP (*dtntcp*) [15]; 4) our TCP exchange generator (*tcpx*) that can produce TCP traffic in user-defined patterns; and 5) a DTN route location and maintenance tool (*dtndctrl*) we have developed to learn about available DTN nodes in the network and dynamically configure DTN links and routes for *dtnd*.

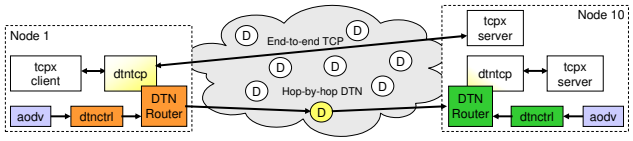

Figure 6: Emulation Setup for DTN-AODV Tests

The basic setup is depicted in figure 6: Our *tcpx* client connects via a fixed port to the co-located *dtntcp* which forwards the data to a pre-configured target, the *tcpx* server. We have modified *dtntcp* so that, upon an incoming connection, it first tries to connect directly to its peer without using DTN routing. Only if the connection setup fails *dtntcp* reverts to DTN routing and forwards its data as bundles. We have not modified the *dtnd* and therefore cannot provide routing hints from the application in-band along with the bundle. Instead, *dtndctrl* obtains available DTN nodes from the local AODV daemon via the file system and supplies the routes via *dtnd*'s socket-based control interface, which are then used by *dtnd* when forwarding bundles using flooding.

We have set up an emulation environment based upon Xen 3.0 (unstable) running virtual Linux machines (Debian, kernel 2.6.16.13, 64 MB RAM each) on an AMD Athlon 64 3800+ Dual Core PC with 1 GB RAM. We have created 10 DTN-AODV-enabled virtual Linux nodes, with node 1 as sender and node 10 as receiver and simulated their motion and wireless interconnection with ns2 2.29 with the *nsemulation* patch 20060202[14] and nam 1.11.

We have measured with two topologies: 1) For initial testing, we used a topology similar to our wireless measurement scenario: a chain of 10 nodes with point-to-point links in between and all nodes running AODV and our DTN extensions. 2) For a simple (proof-of-concept) mobility scenario, we placed the 10 nodes in a field of 2000m$\times$600m, their motion following the random walk model without pauses and a speed of up to 20m/s. In both scenarios, we used the 802.11b model of ns2 with 200m wireless communication range using omnidirectional antennae. To avoid DTN routing loops, (lacking a dynamic routing protocol for *dtnd*) we allowed forwarding from node m to node n only if $n > m$. Node 1 was the sender and node 10 the receiver as depicted in figure 6.

Scenario 1 was mainly used to test the proper functioning of the all components and their interaction with different bundle sizes. In scenario 2, we have performed several test runs to validate our integration approach: we have configured the *tcpx* client to send a sin-

[11] 58400 bytes every 100 ms or 29200 bytes every 50 ms; however, the local link and TCP layer flow control reduced the effective transmission rate to lower values.

[12] http://www.f4dwu.net/ and http://symoon.free.fr/scs/dtn/, with kind support from Laurent Franck and Simon Paillard.

[13] We are using *AODV-UU* from Uppsala University; see http://cote.it.uu.se/AdHoc/AodvUuImpl.

[14] http://www-ivs.cs.uni-magdeburg.de/EuK/forschung/projekte/nse/

gle message of 29200 bytes every 150 s, for which the co-located *dtntcp* uses end-to-end TCP if a path exists and DTN otherwise. DTN bundle lifetimes were configured to 5 min and each emulation run lasted for 30 min. While our limited number of emulation runs at this point does not support a (quantitative) performance analysis, we observed that TCP-based end-to-end delivery dominated (6–9 messages) in this geographically rather small setting but also that DTN-based delivery augmented the former to improve the total delivery ratio (by 1–5 messages), sometimes achieving 100%.

We can summarize our main (qualitative) observations as follows: 1) the approach has proven feasible in this specific scenario, 2) the AODV-based dynamic location of DTN nodes works and the corresponding route configuration for *dtnd* is useful for augmenting DTN routing protocols but also for bootstrapping DTN communication in ad-hoc networks, 3) the performance of DTN-based communication compared to direct communication in scenario 1 depends on the bundle size, as the overhead for smaller bundles is quite large in terms of processing time in the DTN implementation.

6. CONCLUSION AND NEXT STEPS

In this paper, we have explored the idea of enabling applications in MANETs to choose their preferred way of interaction with their peer(s) depending on the alternatives available at a certain point in time. We have presented a set of extensions to AODV enabling it to discover nearby DTN routers during route search and obtain routing hints from them. Combined with AODV information on (non)reachability of a peers and path characteristics, this puts an application into the position to make sensible choices of their communication mechanisms. We have shown through measurements that DTN-based hop-by-hop communication can improve throughput compared to end-to-end TCP so that the choosing DTN does only affect semantics but not the net throughput of an application. While our implementation is still at an early stage, we have conducted basic functional experiments with our prototype in an emulation environment, validating the principle of our approach but also encountering a few specification and interoperability nits.

We are presently enhancing our implementation and improving the integration APIs between AODV and DTN as well as towards the applications (for which we also consider extending *dtnd*). This will provide the basis for (a) expanding our emulation work to study the impact of mobility and different mobility models, (b) investigating the use of different DTN routing protocols in conjunction with AODV, and (c) analyzing the impact of different application decision policies. A further step will be scaling our experiments to larger settings for which we will also look into simulations.

7. REFERENCES

[1] T. Clausen and P. Jacquet. Optimized Link State Routing Protocol (OLSR). RFC 2326, October 2003.

[2] M. Demmer. Protocol for the DTN TCP Convergence Layer. Available via the code Wiki at http://www.dtnrg.org/, 2006.

[3] A. Doria, M. Uden, and D. P. Pandey. Providing connectivity to the saami nomadic community. In *Proceedings of the 2nd International Conference on Open Collaborative Design for Sustainable Development, Bangalore, India*, December 2002.

[4] DTN Ref. Implementation. http://www.dtnrg.org/wiki/Code.

[5] K. Fall. A Delay-Tolerant Network Architecture for Challenged Internets. In *Proceedings of ACM SIGCOMM 2003, Karlsruhe, Germany*, August 2003.

[6] Z. J. Haas and M. R. Pearlman. The Performance of Query Control Schemes for the Zone Routing Protocol. In *Proceedings of ACM SIGCOMM 1998, Vancouver*, 1998.

[7] K. Harras, K. Almeroth, and E. Belding-Royer. Delay tolerant mobile networks (DTMNs): Controlled flooding schemes in sparse mobile networks. In *IFIP Networking*, 2005.

[8] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Pocket Switched Networks and Human Mobility in Conference Environments. In *ACM SIGCOMM Workshop on Delay-Tolerant Networking (WDTN)*, 2005.

[9] S. Jain, K. Fall, and R. Patra. Routing in Delay Tolerant Networks. Proceedings of the ACM SIGCOMM 2004, Portland, OR, USA, 2004.

[10] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet. In *Proceedings of the ASPLOS-X conference, San Jose, CA, USA*, October 2002.

[11] A. Lindgren and A. Doria. Probabilistic Routing Protocol for Intermittently Connected Networks. Internet Draft draft-lindgren-dtnrg-prophet-02, Work in Progress, March 2006.

[12] A. Lindgren, A. Doria, and O. Schelen. Probabilistic routing in intermittently connected networks. In *The First International Workshop on Service Assurance with Partial and Intermittent Resources (SAPIR)*, 2004.

[13] Mindstream project. http://mindstream.watsmore.net/.

[14] O. Mukhtar and J. Ott. Backup and Bypass: Introducing DTN-based Ad-hoc Networking to Mobile Phones. In *Proceedings of RealMAN 2006*, May 2006.

[15] J. Ott and D. Kutscher. Applying DTN to Mobile Internet Access: An Experiment with HTTP. Technical Report TR-TZI-050701, Universität Bremen, July 2005.

[16] J. Ott and D. Kutscher. Bundling the Web: HTTP over DTN. In *Proceedings of WNEPT 2006*, August 2006.

[17] A. Pentland, R. Fletcher, and A. Hasson. DakNet: Rethinking Connectivity in Developing Nations. 37(1):78–83, January 2004.

[18] C. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequences Distance-Vector Routing (DSDV) for Mobile Computers. In *Proceedings of ACM SIGCOMM 1994, London, UK*, 1994.

[19] C. E. Perkins, E. M. Belding-Royer, and S. R. Das. Ad hoc On-Demand Distance Vector (AODV) Routing. Experimental RFC 3561, July 2003.

[20] M. Petrova, L. Wu, M. Wellens, and P. Mähönen. Hop of No Return: Practical Limitations of Wireless Multi-Hop Networking. In *Proceedings of RealMAN 2005*, July 2005.

[21] K. L. Scott and S. C. Burleigh. Bundle Protocol Specification. Internet Draft draft-irtf-dtnrg-bundle-spec-02, Work in progress, September 2005.

[22] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical Report CS-200006, Duke University, April 2000.

[23] Y. Wand and H. Wu. DTN-MSN: The Delay Fault Tolerant Mobile Sensor Network for Pervasive Information Gathering. In *Proceedings of IEEE Infocom 2006, Barcelona, Spain*, April 2006.

[24] P. Yao, E. Krohne, and T. Camp. Performance Comparison of Geocast Routing Protocols for a MANET. In *Proceedings of IEEE Infocom 2002, New York, USA*, 2002.

[25] W. Zhao, M. Ammar, and E. Zegura. A Message Ferrying Approach for Data Delivery in Sparse Mobile Ad Hoc Networks. In *Proceedings of ACM Mobihoc, Tokyo*, 2004.