

Demo Abstract - Ad Hoc Social Networking using MAND

Patrick Stuedi, Oriana Riva, and Gustavo Alonso
Systems Group, Department of Computer Science, ETH Zurich
8092 Zurich, Switzerland
{stuedip,oriva,alonso}@inf.ethz.ch

ABSTRACT

MAND (Mobile Ad Hoc Network Directory) is a middle-ware infrastructure for the distribution, storage, and lookup of key/value pairs in ad hoc networks. The key insight in MAND is to piggyback tuples and requests on the messages that routing protocols exchange to build and maintain routes in the ad hoc network. MAND can be used with different routing protocols and does not require any modification to the routing protocols themselves. Using MAND we have built AdSocial, a social networking application that allows users to exchange messages, set up SIP-based VoIP calls, and communicate common interests with nearby buddies. This demo will show the functionality of AdSocial and how MAND performs in a network of 10-15 Nokia N810 Internet Tablets communicating using AODV or OLSR. Conference participants will be able to interact with AdSocial while moving around thus configuring a mobile and multi-hop ad hoc network.

1. INTRODUCTION

115 million units of smart mobile devices (smart phones and wireless handhelds) were shipped in 2007 and more than 180 million shipments are expected in 2008 [4]. With an annual increase of 60%, this represents one of the fastest growing technology markets. With millions of smart mobile devices daily available in offices, conference venues, train stations, airports, buses, etc., mobile ad hoc network (MANET) applications become an interesting possibility to spontaneously interconnect these devices. In a variety of scenarios, MANETs may represent the only feasible networking solution (e.g., emergency rescue operations) or may act as seamless extensions of conventional infrastructure-based networks. Alternatively, due to cost or simply performance reasons, MANETs may represent a convenient alternative even for supporting traditional Internet-based applications (e.g., social networking applications).

Developing applications over MANETs, however, poses several challenges. A key problem is that their dynamic and

volatile nature makes it unfeasible to rely on any centralized component to provide common directory-based services such as DNS [9], SIP [12], SLP [7], or simple name directory servers. Numerous attempts have been made to adapt such services to MANETs. However, existing solutions are either dependent on the routing protocol [11] or they incur in several limitations on how the MANET can be operated by assuming the presence of a fixed and stable super node [3, 8] or by building overlay networks that cannot be maintained without unreasonable costs [6, 8].

MAND (Mobile Ad hoc Network Directory) solves the aforementioned problem by exploiting the fact that MANET routing protocols constantly exchange routing messages. The networking information required by directory-based services can be simply piggybacked on routing messages. By doing so, MAND provides a message-efficient and routing-independent distributed service for storing and performing lookup of key/value pairs in MANETs.

Using MAND we have implemented a social networking application called *AdSocial*. AdSocial supports profile sharing, instant messaging, and SIP-based VoIP. MAND has been tested in ad hoc testbeds of several dozens of laptops and PDAs. In addition, it has also been used during field trials organized in office environments and during a 4-day trial where it was successfully used by 40+ members of our research group both in indoor and outdoor settings. In the demo, we will demonstrate MAND and AdSocial using 10-15 Nokia N810 Internet Tablets. In the following we give a more detailed description of the MAND architecture and the AdSocial application.

2. MAND OVERVIEW

MAND (Mobile Ad Hoc Network Directory) is a distributed system for storing and searching key/value pairs (tuples) in MANETs (see Figure 1). Both distribution and lookup of tuples is done by piggybacking routing-layer messages. The MAND user space process uses netfilter [1] to intercept routing layer messages right before and after they are processed by the routing module. By doing so, MAND makes sure that the routing protocol itself does not have to be modified or re-compiled. MAND is independent on the routing protocol used. The routing-specific functionality such as piggybacking a tuple onto a given routing message or extracting a tuple from a message are performed by the *routing handler*, which is a plug-in component loaded by MAND at startup time. Currently we have implemented a routing handler for the OLSR [5] proactive routing protocol and for the AODV [10] reactive routing protocol.

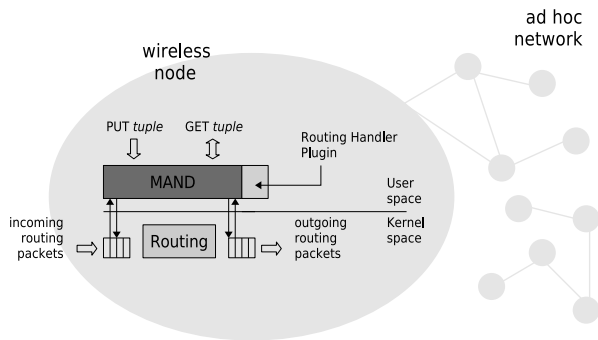


Figure 1. The MAND system architecture

MAND makes use of two basic datatypes, namely *tuple* and *tuple-request*. Tuples are used to map keys to values. Two tuples are said to be equal or match if they have the same key. A tuple contains also a *scope*, a *lifetime*, and a *version* field. The scope field specifies for how many hops at most the tuple must be disseminated in the network. The lifetime field specifies for how long a tuple is stored locally at a node. The version field specifies a replacement schema among tuples with the same key. Tuple-requests have the same fields as tuples, except that there is no value field.

To insert a tuple into the system, an application calls the `put(tuple)` interface. When a tuple enters MAND on a node, its scope value is decremented and if there is any pending request for that tuple a response is sent. The tuple is stored locally if there are no other tuples with the same key or if this has a higher version than any existing tuple with the same key. MAND is constantly listening for incoming routing messages. Once a routing message is intercepted, it is passed to the routing handler which is in charge of piggybacking the tuple to the message. The altered message is then sent to the kernel network stack and eventually leaves the node. On a remote node the routing message with the piggybacked tuple is received, the routing handler extracts the tuple from the message, and the tuple is eventually stored.

To lookup a tuple with a given key, an application calls the `get(tuple-request)` interface. When a tuple-request enters a node, the system first checks whether the requested tuple is in the local memory. Otherwise, the tuple-request is stored locally and a copy of it is passed to the routing handler which will piggyback the tuple-request on an intercepted route request message. The altered route request may enter a node where the requested tuple is stored locally. The routing handler extracts the tuple-request from the message and piggybacks the matching tuple on a route reply which is sent back to the node that issued the route request. If a tuple cannot be found in the network, its corresponding tuple request will remain stored until its lifetime expires. Eventually, a negative acknowledgment will be sent to the application that submitted the request.

MAND has been implemented in C and tested in ad hoc networks of both laptops and Nokia N800/N810 Internet Tablets. Figure 2 shows a network testbed of 23 Tablets.

3. ADSOCIAL

Using MAND we have built AdSocial, a social networking application that resembles Facebook, ICQ, or Flickr and



Figure 2. The MAND testbed consisting of 23 Internet Tablets

runs on MANETs. The key difference between AdSocial and traditional social networks lies in the way users communicate with each other. In Internet-based social applications, users communicate with *friends*, i.e., other previously contacted users. In AdSocial, interactions take place between nearby users in the ad hoc network, so called *buddies*. An ad hoc network might form temporarily in everyday life situations, e.g., on the train when travelling to work, at a conference, or during a business event. An AdSocial user creates a profile in which he/she specifies interests and other personal information. When the user is online, AdSocial notifies the user's presence in the ad hoc network. Users in the ad hoc network can retrieve the profile of their buddies as well as search for nearby buddies with specific interests. AdSocial is designed to support spontaneous interactions and provide the basis for running collaborative applications (e.g., gaming, chatting, VoIP, etc.). A screenshot of the application is shown in Figure 3.

Besides retrieving the profile of nearby buddies, AdSocial supports instant messaging and SIP-based VoIP calls. To run SIP in MANETs, MAND provides a decentralized implementation of the SIP architecture [13]. Each node runs a modified SIP proxy. Each time a user registers with the local SIP proxy, besides internally storing the user's location address, the proxy uses MAND to advertise itself as the contact address for the registered user (i.e., it injects into MAND a tuple whose key is the SIP user name and whose value is the IP::port contact information of the proxy). If a SIP proxy receives a SIP INVITE message for an unregistered user, it consults MAND to lookup the proxy which was advertised as outbound proxy for this user and forwards the INVITE message. As our SIP implementation maintains the same interface and message flow of the standard SIP architecture, it allows any SIP-based application commonly used in the Internet to be used in MANETs without any modification. AdSocial is implemented as a web application. It consists of a web server running as a FastCGI module that uses MAND to discover nearby AdSocial users. The communication with the local browser is realized using AJAX calls. Users' presence announcements are exchanged periodically using MAND. All inserted tuples have a common key and their value contain a URL to the user's web server. Searching for nearby buddies is done by simply looking for tuples with the given key.

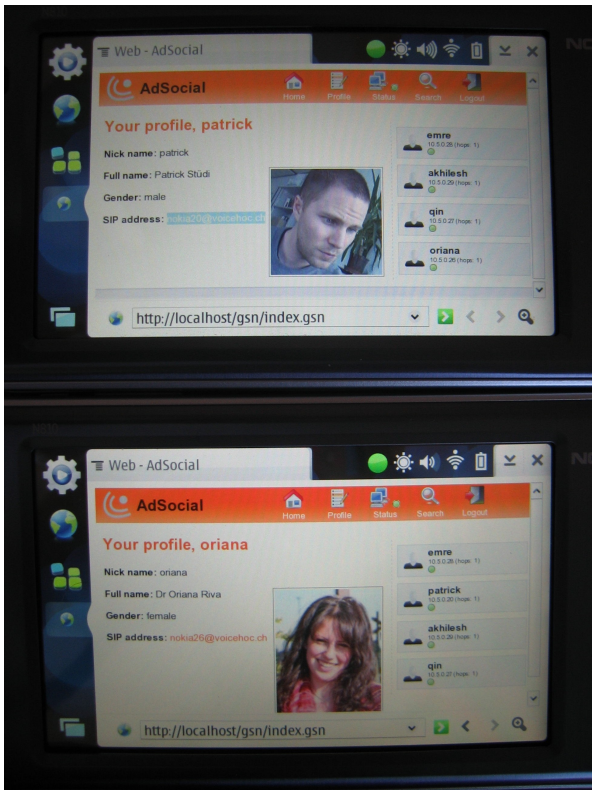


Figure 3. The AdSocial application on two N810 handhelds

4. DEMONSTRATION HIGHLIGHTS

In this demo we use 10-15 Nokia N810 Internet Tablets. Nokia N810 tablets have 400 MHz processor, offer 128 MB of RAM, and run the Internet Tablet OS based on Linux 2.6.18. In the demo, all handhelds run MAND and the AdSocial application. The ad hoc network is established using either AODV or OLSR. We use a transmission power of 10 mW in order to create a multi-hop topology even in a limited space. To limit the network interference the devices are configured to use a frequency channel distant enough from the one used by the available WLANs. Conference participants will be able to interact with AdSocial while residing in different spots of the conference room(s) and possibly moving around. Our demo will include the following highlights.

- *AdSocial application*: we illustrate the AdSocial functionality and give practical insights on the user experience that can be provided in settings such as a crowded conference venue. The AdSocial application particularly fits the conference scenario. Each participant is provided with a N810 Tablet and can freely interact with other users through AdSocial. Each user can quickly create his/her own profile under login/password authentication, and, after publishing his/her interests, he/she can find buddies with matching interests, exchange chat messages, and even establish SIP-based video calls that use the camera integrated in the handheld. Some participants may also use AdSocial outside the demonstration session, e.g., during coffee breaks or social events. AdSocial has already been extensively tested in several indoor and outdoor real-world scenar-

ios such as offices, cities, and bars by 40+ people.

- *Network topology and routing performance*: while AdSocial runs, information on the network topology and routing protocol such as number of neighbors and distance in number of hops can be visualized on the handheld by invoking an additional applet. We show how this information varies in presence of multi-hop communication and mobility. In addition, as our current implementation supports both AODV and OLSR, we can switch from one routing protocol to the other and show how the reactive or proactive approach of AODV and OLSR respectively may impact on the application's responsiveness and communication reliability.
- *Packet-level analysis*: during the demo we have one laptop running the Wireshark [2] packet sniffer tool in order to provide a packet-level analysis of the MAND's piggybacking mechanism and the routing behaviour when setting up routes, maintaining the list of buddies, supporting video calls over SIP, etc. This allows conference participants to understand in more details how MAND and the routing protocol interact.

References

- [1] The Netfilter/Iptables Project, June 2001. <http://www.netfilter.org>.
- [2] Wireshark: The World's Most Popular Network Protocol Analyzer. <http://www.wireshark.org>, 2007.
- [3] F. Araujo, L. Rodrigues, J. Kaiser, C. Liu, and C. Miti-dieri. CHR: A Distributed Hash Table for Wireless Ad Hoc Networks. In *Proceedings of the 4th International Workshop on Distributed Event-Based Systems (DEBS'05)*, pages 407–413. IEEE Computer Society, 2005.
- [4] Canals. <http://www.canals.com/pr/>, February 2008.
- [5] T. Clausen and P. Jacquet. Optimized Link State Routing Protocol (OLSR). RFC 3626, October 2003.
- [6] C. Cramer and T. Fuhrmann. Proximity Neighbor Selection for a DHT in Wireless Multi-Hop Networks. In *Proceedings of the 5th IEEE International Conference on Peer-to-Peer Computing (P2P'05)*, pages 3–10. IEEE Computer Society, 2005.
- [7] E. Guttman, C. Perkins, J. Veizades, and M. Day. Service Location Protocol, Version 2. RFC 2608, June 1999. Updated by RFC 3224.
- [8] R. Kummer, P. Kropf, and P. Felber. Distributed lookup in structured peer-to-peer ad-hoc networks. In *Proceedings of the International Conference on Distributed Objects and Applications (DOA'06)*, volume 4276, pages 1541–1554. LNCS, 2006.
- [9] P. Mockapetris. Domain names: Concepts and facilities. RFC 882, Nov. 1983. Obsoleted by RFCs 1034, 1035, updated by RFC 973.
- [10] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561, July 2003.
- [11] H. Pucha, S. M. Das, and Y. C. Hu. Ekta: An Efficient DHT Substrate for Distributed Applications in Mobile Ad Hoc Networks. In *Proceedings of the 6th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'04)*, pages 163–173. IEEE Computer Society, 2004.
- [12] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261, June 2002. Updated by RFCs 3265, 3853.
- [13] P. Stuedi, M. Bühr, A. Remund, and G. Alonso. SIPHoc: Efficient SIP Middleware for Ad Hoc Networks. In *Proceedings of the 8th ACM International Middleware Conference (Middleware'07)*, pages 60–79, 2007.