A SURVEY ON LUNAR PROTOCOL

Ankush D.Patil, Rima Lingawar

ABSTRACT
The current networking design of the Internet architecture has shown some limitations. Restricted by inherent layering constraints, valuable networking information cannot flow freely inside the network stack and potential operational optimizations are impossible to achieve. To overcome these limitations, we extend the current trend of cross-layer approaches with a framework called *underlay protocol fusion*: the basic building blocks of Internet functionality are factorized out and merged in a function pool where information sharing and operational optimizations are performed.

To illustrate our approach, we present LUNARng (LUNAR next generation). It is a fully distributed underlay protocol designed for the Internet integration of wireless ad hoc net- works (MANETs) where fundamental services such as name resolution, address autoconfiguration, and IPv4/IPv6 routing are transparently available whether the MANET is connected or not to the Internet. Internet integration refers here to the ability to *insert/remove* a MANET *into/from* the logical organization of the Internet without any loss of functionality. Moreover by using *protocol models*, the underlay nature of LUNARng allows to optimally merge (with respect to the multi-hop nature of MANETs) network operations which are traditionally carried out at different layers of the protocol stack.

Keywords: Cross layering, DNS, IPv4, Internet Integration, Protocol Fusion, Underlay MANET.

1. INTRODUCTION:

1.1 Internet Integration

The design of the Internet is based on thirty years oldlayering approach which aimed at factoring outfunctionality. Networking concepts were sought which are able to stretch from local scale to global size, from slow links to highspeed trunks, from PDAs to supercomputers. In spite of its slow evolution and monolithical design, the "canonical set" of protocols collected in the Internet-Suite has done a surprisingly good job during the last decade. Today however, the limitations of this one-size-fits-all approach have become visible especially with the advent of wireless communications.

These limitations can be linked to the incapacity of the Internet to scale in a functional way at the networking layer. It has evolved through a patch style which has not added variety: additions were made in stealth way and have not dared to radically change or extend the core of IP forwarding. We refer here to the introduction of the hidden routing hierarchy and mechanisms of AS, CIDR or MPLS as well as other less successful projects (in terms of large-scale deployment) like RSVP, IP Multicast, and Mobile IP.Due to the end-to- end principle, the place where the Internet has envisaged and endorsed functional scaling, that is the possibility to freely add arbitrary customized functionality, is the application layer. This is whereremarkable breakthroughs have been achieved and variety was obtained: DNS, email, Web, VPN, VoIP and P2P are the highlights to mention here.The low flexibility of the current Internet protocol suite is particularly striking when considering mobile wireless ad hoc networks (MANETs). The inherent distributed and infrastructure-less nature of MANETs has indeed highlighted how fundamental services of the Internet rely on a centralized client-server model. That is, the absence of basic services such as name resolution or address allocation does not strictly prevent networking, but it strongly restrains the adoption of wireless ad hoc networking as a plug-and-play technology for the masses.

Therefore the ability of an autonomous MANET to exhibit Internet-like functionalities (while not being connected to the Internet) is one facet of Internet integration: users should not experience a loss of commodity other than the loss of global connectivity. A complementing aspect is the seamless integration of (mesh) MANETs with the logical (e.g. global addressing) and operational (e.g. name resolution) organization of the Internet. This property is the second facet of Internet integration: the ability for a MANET to adapt its internal behavior in order to *insert* itself into a larger organization such as the Internet.

1.2 Underlay design for MANET

As stated earlier, the strict layering approach of the existing legacy TCP/IP model is slowing down or even preventing innovative functionalities to appear at the network layer. It is commonly agreed that more inter-layer coordination is needed in wireless networks when considering on one side the network layer, and on the other side the physical and the link layers [1]. However, more coordination is also required among higher layer protocols. For example, we already demonstrated in [2]
How a single DNS name resolution request procedure cangather enough *cross-protocol* data to fulfill the tasks of linklayerresolution and path setup.In order to achieve such optimizations, we introduce anunderlay shim that performs *protocol fusion* based on *protocolmodels*. The goal is first to gather the previously isolated
Information provided by different task-specific protocols of thenetwork stack, and second to optimize the operation of theseprotocols by anticipating their needs. In contrast to more classicalcross-layer techniques which provide hints and triggersbetween layers, we use an underlay located between the IPand Ethernet layers in order to have full control over the datacoming in and out of a node. Tasks that were previously

carriedout by remote servers at different layers are now performed atLayer 2.5 (i.e. hence the name *underlay*), and the historicalbarriers between the somehow isolated protocols involvedat the network layer are suppressed. As a result, effectiveoptimizations can now be achieved.

Moreover, to realize thetwo facets of Internet integration, the functionalities providedat layer 2.5 are activated on-demand when the MANET is notconnected to the Internet, or they can be bypassed depending ifa given service is provided by the infrastructure-based networkwhich provides the global connectivity.The paper is organized as follows. In the next section, wesituate our approach with respect to traditional cross-layerschemes and we introduce our underlay technique. We thenbriefly introduce LUNARng and summarize the features itprovides. We then describe the mechanisms that we use toperform the Internet integration of wireless ad hoc networks,and we also detail the concept of protocol fusion. We also present some implementation details of our approach whichhas been successfully validated and deployed on a real tested.Finally, we conclude with a discussion of future open researchchallenges.

Ankush D. Patil is currently pursuing master's degree program in Computer Science and Engginering in Sagar Institute of Research & Technology, Bhopal, MP, India.
E-mail:- ankushpatil48@gmail.com
Rima Lingawar is currently pursuing master's degree program in Computer Science and Engginering Jagdambha college of Engineering & Technlogy, Yavatmal, Maharashtra, India.

2,CROSS LAYERING VS. PROTOCOL FUSION :

1.1 Traditional cross-layer design

Cross-layer design is an active field of research which sofar has not looked into Internet integration issues. Insteadthe focus is on *wireless* networking, mainly because manyprotocols and services of the *wired* Internet are inefficientin the presence of unreliable wireless links and unpredictable topological changes. Without cross-layer design, the existinglayer boundaries unfortunately prevent the development ofpotential optimizations1. Since the differences between wiredand wireless networks lie in the physical and link layers, i.e.the layers located below the IP layer, a large majority ofcross-layer techniques concentrates on providing lower-layerfeedback to the network layer [1] (e.g. to notify link-layerevents such as layer 2 handovers) and, to a smaller extent, tothe transport layer (e.g. to achieve TCP performance optimizationsvia wireless-specific fine-tuning of TCP parameters). We denote such approaches as lower-layer feedback techniques, and illustrate them by Fig. 1 where valuable information is passed from the lower-layers to the higher layers, which cansubsequently optimize their operation. The main advantage ofthis approach is that it becomes possible to re-design a givenprotocol such that it specifically reacts to some lower-layersignals, i.e. the protocol becomes aware of what happens inlower-layers. However, such lower-layer awareness does notcome without a cost: it requires code modifications insidethe protocol stack which can restrict the possible wide-scaleadoption of such changes.

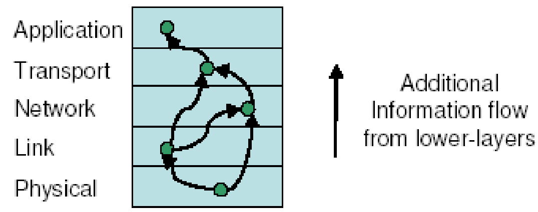

Fig.1. Lower-layer feedback

As a consequence, most cross-layer schemes remain research prototypes. It is also worth mentioningthat cross-layer designs using an *information-bus* (i.e. aTransversal layer that receives/sends specific feedback from/toall layers via specific function calls) unfortunately suffer fromthe same implementation and deployment restrictions.

2.2. Case study: the failure of name resolution

In the Internet, name resolution is performed via the DomainName System (DNS) which relies on a hierarchy of serversdistributed around the world. One issue is that dynamic nameresolution in traditional IP networks assumes that there existsa reachable DNS server at all time: the whole operation ofnameresolution collapses if no server is available. All existing
Operating systems do not even try to send a name request ifno DNS server is configured in the system: if a node is notconfigured with a DNS server address, it simply assumes thatdynamic name resolution is not available. The implementationof a DNS-compatible name resolution system in a MANET istherefore challenging in many ways.Actually, a *natural* way of performing name resolution in aMANET is to use a decentralized approach in which a node ofthe MANET replies to a broadcasted name request for which itis the target. Different flavors [2][3][4] of such an approach canbe found in the literature. Although the operation of distributedname resolution resembles the route discovery procedure ofa reactive routing protocol, it is more difficult to implementthan routing since the DNS operation is *hard coded* in currentoperating systems and applications. That is, by default, anode configured with a DNS server address sends its unicastDNS request messages via the network interface towards theserver. In a MANET, this procedure becomes irrelevant andit should be replaced with a MANET specific mechanism.

ISSN 2229-5518

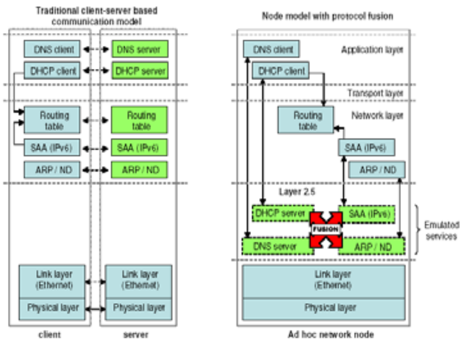

Fig.2. Protocol emulation and fusion

Moreover, a name resolution scheme for MANETs should notprevent a node from resolving names in the classical way whenthe MANET is connected to ainfrastructure-based network.In particular, the existing APIs and related protocols shouldremain unchanged since it is not conceivable to modify thehuge amount of existing applications such as web browsersand email clients.

2.3. Internet integration (part I): service emulation with models

As introduced earlier, our approach relies on an underlayprotocol located below the IP layer and above the Ethernetlayer. Being located below IP, the underlay protocol is awareof all traffic coming in and out of a node. By manipulatingthe stream of messages passing by, it becomes possible toperform some *cross-protocol* optimizations. By cross-protocoloptimizations we mean that, among other options it is possibleto merge the operation of independent protocols by anticipatingthe needs of a given protocol: the global behaviorof a TCP/IP-based protocol stack is indeed well known. Forexample, with a reactive routing protocol for wireless ad -hocnetworks (such as AODV [5] and DSR [6]), a successfulroute request (RREQ) procedure will always be followed bya link-layer resolution request for the next hop node towardsthe destination. One can therefore design the route requestprocedure such that it also performs the anticipated link-layeraddress resolution. A main advantage of using an underlayprotocol is that it requires no modifications to the existingprotocol stack. Furthermore, protocol fusion is invisible tolegacy layers: it is thus possible to *fool* the higher layers bysending appropriate control messages to the protocols locatedin both the network and the application layers. It thus becomespossible to build *protocol models* which mimic the behaviorof some specific network functionality in order to provideInternet-like services while the MANET is not connected tothe Internet.

We define this property of Internet integration as*service emulation*. One can note that the underlay can also beused to hide the multi-hop nature of a wireless ad hoc networksuch that the IP stack believes that the node is connected to aclassical IP subnet on a single layer 2 link.The concept of protocol emulation and fusion is illustratedby Fig. 2. The left side of the figure shows the client-servermodel used in traditional Internet-like networking. Fundamentalservices such as domain name resolution (DNS) andaddress autoconfiguration (DHCP for IPv4/IPv6, and SAA [7]for IPv6) fully rely on the presence of a dedicated server.Other protocols such as ARP and ND [8] are not based on theclient-server model but for optimization purposes (as describedlater) their operation is also preempted by the underlay. Inthe right part of Fig. 2, we illustrate the underlay-basedmodels located at layer 2.5.

In order to overcome the absenceof legacy servers, models of the basic Internet services areimplemented in the underlay: these *emulated functionalities*are activated on-demand when required. Moreover, at theunderlay level it is now possible to optimize the operationof protocols which were previously opaque to eachother. Withthis protocol fusion, network operations become optimized tothe topological and operational properties of wireless ad hoc network.

3. LUNARNG:

To explore and demonstrate the feasibility of underlayfusion, protocol emulation, and Internet integration, we haveextended the original operation of LUNAR [9], i.e. a reactiverouting protocol initially designed to back up the developmentof network pointers [10]. As for the features provided by theprotocol fusion in the underlay, our next generation LUNAR(LUNARng) combines IPv4 and IPv6 path setup, link-layeraddress resolution (ARP/ND), and name resolution in a singlerequest/reply operation optimized for distributed wireless adhocenvironments.

Moreover, thanks to the use of networkPointers as the basic forwarding abstraction, anIPv6 multihopdata path can include nodes which are only IPv4 enabled(and vice-versa).The two facets of Internet integration are also covered,since LUNARng provides Internet-like services via protocolemulation when the MANET is autonomous and Internetadaptation via coherent global addressing, routing, and nameresolution when the MANET is connected to the Internet.

Moreover Internet integration is transparent to the MANETusers, in the sense that they only witness the appearance orloss of global connectivity.In practice, LUNAR is implemented as a Linux kernelmodule2 that can be dynamically loaded on a host and whichrequires no single modification to the Linux kernel code.LUNAR positions itself between the IP and Ethernet layers(actually just above the wireless device driver) and createsa subnet illusion with respect to the IP stack. Upon startup,the LUNAR module creates a virtual network interface thatis internally linked with the real wireless interface connectedto the MANET. Hence, all traffic that flows via the virtualinterface is seen by the LUNAR module which can specifically react to particular messages.

4. INTERNET INTEGRATION IMPLEMENTED

In this section, we describe the mechanisms developedin LUNARng in order to perform Internetintegration andoptimize the operation of Internet-like protocols with respectto distributed wireless networking.

4.1. Filling expectations and building models

As stated previously, the basic steps and behavior of

acommunication startup with the TCP/IP protocol stack is wellknown. In a wired network, the initiation of a communicationusually conforms to the following steps (we assume that eachstep is successful):
1. The user specifies the name of the host s/hewishes to communicate with,
2. The IP stack triggers an ARP/ND (IPv4 AddressResolution Protocol or IPv6 Neighbor Discovery)request in order to resolve the link-layeraddress of thenext hop towards the DNS server(or of the DNS server itself),
3. The system sends a DNS request to resolve the target host name,
4. The IP stack sends an ARP/ND request in order toresolve the link-layer address of the next hoptowards the target (or of the target itself).

In a MANET that uses a reactive routing protocol, if weassumethat a node is configured with a DNS serveraddress,and if we assume that there exists a DNS server in theMANET, the following steps are executed:
1. The user specifies the name of the host s/hewishes to communicate with,
2. The MANET routing module triggers a route request (RREQ) procedure to find a path to the DNS server,
3. The IP stack triggers an ARP/ND request in orderto resolve the link-layer address of the next hoptowards the DNS server (or of the server itself),
4. The system sends a DNS request to resolve the IPaddress of the target host name,
5. The MANET routing module triggers a RREQ/RREPprocedure to find a path to the IP address of thetarget host,
6. The IP stack sends an ARP/ND request in order toresolve the link-layer address of the next hoptowards the target (or of the target itself).

It is clear that the specificities of MANETs already increasethe total overhead because routes have to be discovered ondemand.Moreover, we assumed here that there existed aDNS server, but this assumption will usually be false in realinfrastructure-less MANETs. Hence to resolve the specificissues introduced by the distributed and autonomous nature ofwireless ad hoc networks, we introduce an optimized schemewhich specifically considers and addresses the particularfeaturesof MANETs.

4.2 Revisiting route requests

The key element of our underlay approach is that in a MANET, one can trigger the route request procedure with the**name** of the destination host rather than with its IP address.At the same time, if the RREQ procedure can gather enoughinformation, the number of required steps can now be greatlyreduced:
1. The user specifies the name of the host s/hewishes to communicate with,
2. The MANET routing module triggers a route requestRREQ/RREP procedure to find a path to thespecified target host name. The RREP eventuallycontains the target IP address (es) and the linklayeraddress of the next hop towards the target.

Once the originating host receives this information bundle,it knows all relevant details to answer subsequent informationrequests (ARP) internally, and the communication can startimmediately. The path discovery procedure is thus triggered bythe name request message, i.e. an application layer protocol.
The route is then discovered (network layer), and the linklayeraddress is resolved at the same time (network and linklayers). Our underlay approach perfectly suits to the aboveoptimization since we can capture the initial DNS request andtranslate it to an appropriate RREQ message: with a singleRREQ/RREP procedure, we have performed name resolution, path setup, and link-layer address resolution.

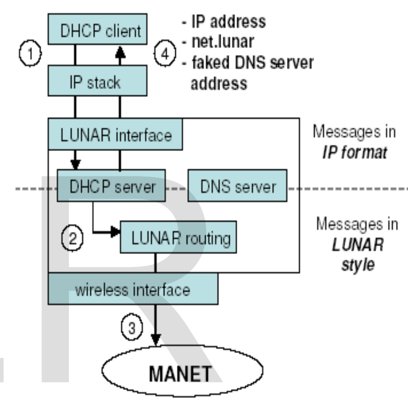

Fig.3. LUNAR DHCP operation

4.3. Internet integration (part II): Internet adaptation

On top of the merging of resolution requests, LUNARng supports address autoconfiguration by implementing a virtualDHCP server which assigns the IP address to the virtualinterface, should the user want to automatically configure thisinterface via DHCP. This mechanism is illustrated by Fig. 3.In step 1 of Fig. 3, a DHCP client sends a request towardsthe LUNAR interface. This request is intercepted by theDHCP engine of LUNAR which randomly chooses an addresswithin a predefined LUNAR subnet (e.g. 192.168.42.0/24),and which then checks the uniqueness of this address by tryingto build a path towards this address (steps 2 and 3). If thepath setup fails (i.e. indicating that the address is not used),a faked DHCP message is sent to the dhcpclient application(step 4). This message includes the IP address to be used onthis interface, the *net.lunar* domain name, and the address ofa faked DNS server reachable via the LUNAR interface.
We also use a similar mechanism in order to performIPv6 stateless address autoconfiguration (SAA [7]). LUNARintercepts router solicitation (RS) messages sent by the IPstack, and returns back a faked router advertisement (RA)message which contains a MANET-global prefix3.Moreover, in order to perform Internet adaptation when theMANET is connected to the

world-wide network, LUNARngalso supports global address autoconfiguration based on *prefixcontinuity* [11]. That is when there exists a gateway to theInternet,LUNARng can coherently distribute a topologicallycorrect and globally route able prefix to the nodes of the
MANET. The subnet illusion is maintained, and IPv6 multi-homingis also possible (if multiple gateways are present).
Moreover, DNS requests for targets which are not inside
the MANET are forwarded to the gateway: this is madepossible since we introduce a virtual namespace as describedin subsection IV-E.

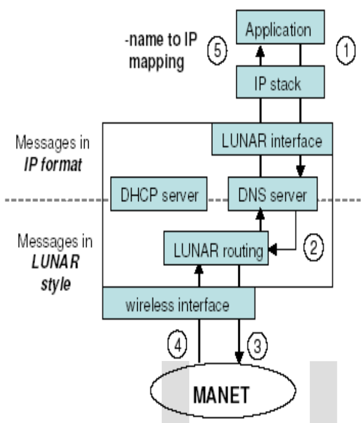

Fig.4. LUNAR DNS operation

4.4. DNS operation

The interception of a DNS request is illustrated by Fig4.
In step 1 of Fig. 4, an application triggers the sending of aDNS request that is intercepted by the LUNAR DNS engine.This triggers a route request procedure which uses the nameof the target to identify the expected destination (steps 2 and3). When the target discovers its name in the route requestmessage, it sends back a route reply message which containsits IP address (steps 4 and 5). Note that this message alsocontains the MACaddress of the next hop node towards thetarget destination: the node which triggered the name resolutionrequest therefore also implicitly performs the link-layerresolution usually carried out by the ARP and ND protocols.
The LUNAR module can then send back a classical DNSreply message to the application which eventually learns the IPaddress of the target. At that point, the network path is alreadyestablished and the link-layer address of the next hop nodeis already known. When the IP stack subsequently issues anARP or neighbor discovery (ND) request, the LUNAR module,which also intercepts these messages, can reply immediatelywithout sending out any messages on the network.

4.5. Net.lunar

The *net.lunar* domain name is configured at startup by the-LUNAR module as being the defaultdomain of a MANETnode. Hence, a hostname request (i.e. not fully qualified) istransformed by the operating system into a net.lunar requestwhich is recognized by the LUNAR module as being aMANETnameresolution. In this way it becomes possibleto identify a simple hostname lookup within the MANETif the user/application only specifies a hostname (e.g. therequest for *cjelger* becomes a request for *cjelger.net.lunar*).

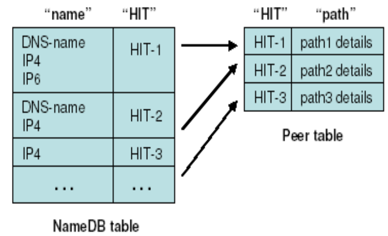

Fig.5. NameDB and peer tables

In other words, a user can easily express its desire to trigger aname resolution inside the MANET. In contrast to host-namerequests, FQDN (fully qualified domain name) requests (e.g.informatik.unibas.ch) are left unchanged by the operat-ingsystem and the LUNAR module will recognize them as being arequest for a host located outside the MANET. If the MANET is connected to the Internet via a gateway, non *net.lunar*DNS requests can be forwarded to the gateway which willpotentially contact a traditional DNS server.

4.6. "Name" cache and path discovery

In order to avoid unnecessary path discovery procedures,
EachLUNAR node maintains a "name" cache, the so calledNameDB (Database) table as shown by Fig. 5. This table-contains all the known identifiers of a given correspondent(also called a *peer*): a DNS-name, one IPv4 and possiblymultiple IPv6 addresses, and a host identifier tag or HITinspired from the Host Identity Protocol (HIP [12]). As aHIT we use a random 128-bit string which becomes the entitywhich LUNAR uses to re-establish paths to a peer. That is,with either name resolution or plain ARP or ND resolution,we bind a peer's name and address to its HIT. All subsequentpath lookup will be carried out with the HIT: as long as ourname cache contains an entry for a peer, we will address thispeer using its HIT. In order to populate the NameDB table,we use the LUNAR route discovery procedure to obtain theidentifiers of a given target, as illustrated by Fig. 6.Moreover, each node can gratuitously populate its NameDBtable by overhearing the information contained in the RREQmessages it forwards. Also, a quiet node which does notsend RREQ messages can also send unsolicited HELLOmessages in order to notify the network about its existence.A user can thus learn the names of other computers connectedto the MANET, since a summary of the informationcontained in the NameDB table is available via the Linux/proc/net/lunar file, illustrated below by Fig. 7.
On thisfigure we can see the *emulated* DNS server peer, the localhost*baobab* which is IPv4 and IPv6 enabled, the peer *mango* forwhich a path (IPv4 and IPv6) is active (peer is resolved), theIPv6-only gateway *banana*,and the peer *cactus* for which a and the peer *cactus* for which a and the peer *cactus* for which aRREQ procedure has been started (i.e. search state).

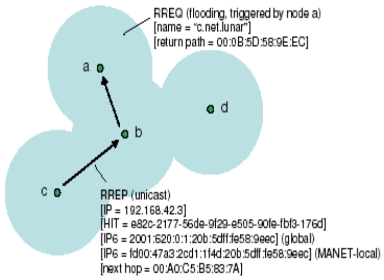

Fig.6. RREQ/RREP procedure

```
# lunar (32 table entries, 4 peers)

FLAGS: N=name, H=hit, 4=IPv4, 6=IPv6,
       r=resolved, s=search, G=gateway

baobab.net.lunar    NH46..    -> Localhost
dns.net.lunar       N.4.r.    -> No lifetime
mango.net.lunar     NH46r.    -> Last heard 6s ago
banana.net.lunar    NH.6rG    -> Last heard 13s ago
cactus.net.lunar    N...s.    -> No lifetime
```

Fig. 7. The proc/net/lunar file.

Note thatfor all resolved nodes the HIT is always known.When possible and to avoid bandwidth waste, LUNARalso uses the NameDB table in order to reply to DNS-PTRrequests for the net.lunar domain. We remind that the goal ofthis *inverse resolution* procedure is to obtain the DNS nameassociated with a given IP address. One must note that withthe traditional DNS operation, a host will send a PTR requestto its DNS server even if it just resolved the IP addressof the corresponding name. This additional overhead occursbecause current operating systems do not implement a namecache and therefore a previously resolved *name! IP address*-mapping cannot be re-used to perform the inverse resolution.

In contrast, if a net.lunar name has recently been resolvedinto the corresponding IP address by a given node N, noPTR request is sent into the MANET if the node N wants toresolve this IP address into a name, as we use the NameDBtable as a cache. Finally, to cope with the volatility of ad hocnetworks, the name cache is periodically drained in order tohandle address or name changes.

5. OPEN ISSUES:

An open issue is the case of two nodes picking identicalhost-names in their FQDN. For example, two nodesrespectively named *john.domain1.net* and*john.domain2.com*will both end up being identifiedinside the MANET as*john.net.lunar*. To resolve this issue, we plan to add a mechanismto check for duplicate names at the same time when wecheck for duplicate IP addresses i.e., with the same RREQmessage. Similar to picking another random IP address incase of a collision, LUNAR will start adding a suffix to thehostname and test again with the new name.

For the previousexample, the two nodes would thus end up being for examplenamed *john.net.lunar* and *john33.net.lunar*.Note that the new name is only used inside LUNAR i.e., atlayer 2.5 and is mostly relevant to the *other* MANET nodestrying to contact the node with the new name: No attempts aremade to change the host's original FQDN (*john.domain2.com*),which (a) would be a challenging implementation exercise and (b) could also be an unwanted source of confusion to theend user. In other words: we keep a mapping table between thenew names and the LUNAR IP addresses, not the old (derivedfrom FQDN) names and the old IP addresses. To allow usersto distinguish between *john* and *john33* which with very highprobability should have different HITs, a user could use the/proc/net/lunar file to distinguish the two nodes if theHIT is derived from a well-known public key identity (see[12]).

A second issue relates to the case of merging networkclouds: This can lead to a MANET where some hosts haveidentical IP addresses4 and/or identical LUNAR names. We(already) solve this problem by introducing stealth "hostidentifier tags" (HITs): any node joining the network with acolliding name or IP address will not be discovered by theLUNAR path establishment procedure as it has a differentHIT. This strategy, which was proposed by [13] and whichis inspired from [12], permits to maintain TCP connectionsalthough new hosts appeared in the MANET with the sameIP address. However, we still need to implement a scheme to perform a large-scale IP(v4) renumbering.

6. CONCLUSION:

In this paper, we have shown and demonstrated how a MANET network can exhibit full Internet integration, thanksto the use of a dedicated underlay scheme which positionsitself just below the IP layer. A key insight of this paper isthat this underlay has to incorporate a model of the IP stack-that it serves, should it wish to create a perfect fixed Internetillusion for it.

In addition to Internet integration, the underlay scheme alsoallows to optimize network operations with respect to thespecific properties of distributed wireless ad hoc networking.This protocol fusion permits to merge the network operationsof name resolution, link-layer address resolution, and network-path setup in a single and efficient procedure.

In particularthis is done without any modifications of the existing operatingsystems, applications, and name resolver library. Since MANETs break several implicit assumptions (like client-serverorientation) of IP networking, our underlay approach permitsto rearrange basic functionalities in a MANET-friendly way. Itis a first step towards a functional recomposition of IP-relatedprotocols outside layering constraints.

REFERENCES

[1] S. Shakkottai, T. Rappaport, and P. Karlsson,"Cross-Layer Design forWireless Networks," *IEEE Comm. Mag.*, vol. 41, no. 10, pp. 74–80,October 2003.

[2] C. Jelger and C. Tschudin,Underlay Fusion of NS, ARP/ND, andPath Resolution in MANETs,"in Proceedings*of ADHOC'05*, May 2005,Stockholm, Sweden.

[3] P. Engelstad, D. V. Thanh, and G. Egeland, "Name Resolution in On-Demand MANETs and over External IP Networks," in *Proceedings ofIEEE ICC'03*, May 2003, Anchorage, Alaska.
[4] J. Jeong, J. Park, and H. Kim, "Name Service in IPv6 Mobile Ad-hocNetwork connected to the Internet," in *Proceedings of IEEE PIMRC'03*,Sept. 2003, Beijing, China.
[5] C. Perkins, E. Belding-Royer, and S. Das, "RFC 3561 - Ad hoc On-Demand Distance Vector (AODV) Routing," July 2003.
[6] D. Johnson, D. Maltz, and Y.-C. Hu,"Internet Draft - The DynamicSource Routing Protocol for Mobile Ad hoc Networks (DSR), draftietf manet-dsr-10.txt," July 2004.
[7] S. Thomson and T. Narten, "RFC-2462 - IPv6 Stateless AddressAutoconfiguration," December 1998.
[8] T. Narten, E. Nordmark, and W. Simpson, "RFC 2461 – NeighborDiscovery for IP Version 6 (IPv6)," December 1998.
[9] C. Tschudin, R. Gold, O. Rensfeld, and O. Wibling, "LUNAR – ALightweight Underlay Network Ad-Hoc Routing Protocol and Implementation,"in *Proceedings of NEW2AN'04*, February 2004, St. Petersburg,Russia.
[10] C. Tschudin and R. Gold, "Network Pointers," in *Proceedings of FirstACM Workshop on Hot Topics in Networks (HotNets-I)*, October 2002,Princeton, NJ, USA.
[11] C. Jelger and T. Noel, "Proactive Address Autoconfiguration and PrefixContinuity in IPv6 Hybrid Ad Hoc Networks," in *Proceedings of IEEESECON'05*, September 2005, Santa Clara, CA, USA.
[12] R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson, "Internet Draft- Host Identity Protocol, draft-ietf-hip-base-03.txt," June 2005.
[13] N. Vaidya, "Duplicate Address Detection in Mobile Ad Hoc Networks,"in *Proceedings of ACM Mobihoc'02*, June 2002, Lausanne, Switzerland.