6-1998

Location Independent Names for Nomadic Computers

David Steere

Mark Morrissey
Portland State University

Peter Geib

Calton Pu

Jonathan Walpole
Portland State University

Citation Details

Steere, David C., Mark Morrissey, Peter Geib, Calton Pu, and Jonathan Walpole. "Location Independent Names for Nomadic Computers." Dept. of Computer Science and Engineering Oregon Graduate Institute (1998).

Location Independent Names for Nomadic Computers

David C. Steere, Mark Morrissey, Peter Geib, Calton Pu, and Jonathan Walpole
Department of Computer Science and Engineering
Oregon Graduate Institute

Abstract

Recent advances in the Domain Name System (DNS) and the Dynamic Host Configuration Protocol (DHCP) have enabled a new approach to supporting mobile users: location independent naming. *In this approach, machines use the same hostname from any internet location, but use an IP address that corresponds to their current location. We describe a protocol that implements location independent naming for nomadic computers, i.e., machines that do not need transparent mobility. Our protocol allows hosts to move across security domains, uses existing protocols, and preserves existing trust relationships. Therefore, it preserves the performance and security of normal IP for nomadic computers at the expense of not providing the transparent mobility of Mobile IP. We contend that this is a reasonable tradeoff for nomadic computing.*

1 Introduction

In this paper we propose location independent naming as a mechanism to support nomadic computing on the Internet. Nomadic computing is a limited, but common form of mobile computing. Nomadic users compute from different locations, but do not require network connectivity while they are moving. For example, a salesman will travel from customer to customer, using his laptop at each location but storing it while driving to the next location.

While Mobile IP was designed for continuously moving computers, it can also support nomadic users. Mobile IP allows users to maintain existing connections while travelling between networks by allowing machines to carry their IP address with them when they move to a new network. Unfortunately, preserving IP addresses across networks introduces several drawbacks, including the performance penalties of triangle routing, the security problems of IP tunnelling through firewalls, and the loss of connectivity due to packet loss from source address filters. The drawbacks are inherent to Mobile IP since it breaks the one-to-one mapping between IP address and network location that is used by the Internet to route packets to the correct destination.

Our approach, location independent naming (LIN), allows a machine to keep the same name as it moves around the Internet by rebinding its name to its local address when it moves. Once this binding has been made, the machine can communicate with any other host on the Internet using standard IP routing, since the source address on its packets identifies the machine's actual location. Further, other machines on the Internet can communicate with this machine, since its hostname maps to its current address. While this machine remains at its current location, it behaves just like any other machine at that location – no special support is needed except to setup and tear down the name-to-address binding. Because LIN preserves the association between IP address and location, it avoids the performance and security drawbacks of Mobile IP for nomadic computers, as well as the complexities of optimizing Mobile IP.

This project was supported in part by DARPA contracts/grants N66001-97-C-8522, N66001-97-C-8523, and F19628-95-C-0193, and by Tektronix, Inc. and Intel Corporation.

LIN represents an advance of the state of the art as it allows correspondent hosts to communicate with a nomadic host using its well-known name without the performance penalties, security issues, or need for infrastructure support of Mobile IP. It leverages and extends existing and proposed functionality of DNS (Domain Name System) and DHCP (Dynamic Host Configuration Protocol) to allow a mobile host to keep its name while moving across security domains using existing trust relationships. LIN achieves these significant advantages by sacrificing the ability to maintain connections transparently while moving from one network to the next. Instead, it explicitly updates the hostname-to-address mapping stored in DNS. We believe this is a useful tradeoff, since many mobile users are nomadic for practical considerations independent of network support (e.g., vibrations and environmental variables), and therefore do not need the continuous connectivity of Mobile IP.

The remainder of this paper describes our approach in more detail. Section 2 describes nomadic computing and argues why it is a common form of mobility. Section 3 presents our protocol, and Section 4 discusses implications and technical details of the protocol. Section 5 presents extensions to our protocol to address some potential shortcomings. Section 6 presents a summary of related work.

2 A case for nomadic computing

Our motivation for this work is somewhat selfish: we want to use our laptops when we travel. Currently, most places we visit are willing to give us IP access but are not willing to bind that address to our machine's name. As a result, we must either not share our laptop's resources or tell the new name to those with whom we share the machine's resources. In addition, our home network is protected by a firewall that prevents us from using IP encapsulation (tunneling), and none of the hosts with which we want to communicate have the support for mobility built into their IP layer as proposed by Chesire and Baker [7] or Myles et al. [17]. As a result, we were forced to look for another alternative.

In attempting to address our need, we realized that nomadic computing is a common form of mobile computing. We base our claim on several observations. First, most network traffic is either connectionless, such as Sun's Network File System, or oriented around short-lived continuously active connections, such as HTTP traffic. For nomadic hosts, the time to move between networks is typically much longer than the lifetime of these connections, and hence users of these protocols have no special need for transparent mobility.

Second, Mobile IP achieves transparent mobility because the relocation to the new network happens atomically with respect to the connection. Atomic relocation occurs either because no packets are sent while the host is disconnected, or because the travelling host is continuously connected, i.e. on a wireless network. Many protocols, such as streaming video, cannot be transparently suspended while the host is moving and hence do not receive the benefit of transparency.

In addition, only a small fraction of Internet hosts are currently connected via wireless networks, and few wireless hosts have the ability or need to change their network location. On one hand, most laptop users run Windows 95/NT/CE from Microsoft, all of which turn off the network interface card (NIC) when the laptop is suspended, automatically terminating any existing connec-

tions. Most laptops are suspended when the lid is closed, or the machine sits idle for several minutes, and thus the likelihood of losing connections while driving or waiting for an airplane to take off is high. On the other hand, most large-area wireless networks (such as Metrocom's Ricochet network [25] or CMU's wavelan network [3]) are represented as a single network, which means hosts maintain the same IP address as they move within the coverage area.[1] Users of cellular modems use the telephone network to dial into an ISP, and hence can maintain the same IP address and network location regardless of their physical location.

Third, there is an inherent risk to true mobility (computing while moving) [24]. This risk involves loss of the laptop due to theft or accident, loss of data from power failure, and poor connectivity while moving. As a result, one would be foolhardy to put critical services on a mobile host due to the high risk of data loss or compromise, and the relatively poor connectivity of wireless hosts make them ill-suited to host heavily used services. Thus most nomadic computers are personal machines, and see relatively low amounts of network traffic.

3 Our approach to location independent naming

Our goal is to provide a means of supporting nomadic users that avoids the performance and security issues of Mobile IP, that requires as little modification to existing protocols as possible, and that can be easily deployed. Our protocol makes use of existing DNS and dynamic host configuration protocol (DCHP) standards, but does require extensions to the travelling host, the addition of an option to DHCP, and a small change to the home DNS server and its secondaries. We believe these changes to be reasonable since the main changes are borne by the travelling host[2](TH), which directly benefits from these changes. The changes to DHCP and DNS are very small, and could be packaged as experimental extensions to the main protocols.

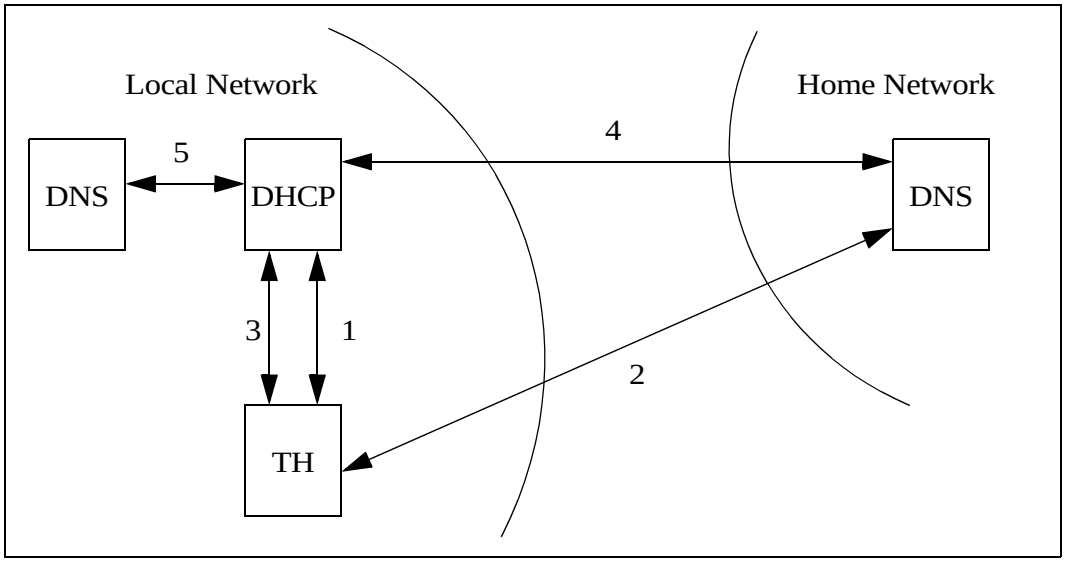
For our purposes, nomadic hosts are personal portable computers. Nomadic hosts can move between networks, but do not require atomic movement and do not move continuously. Usually the time to move between networks is large, on the order of tens of minutes or more, because the user must physically move to a new location and reconnect to a network. Nomadic hosts typically initiate most of their communication (e.g., downloading web pages or e-mail), but they may export resources such as files to a few other clients. In order to access these resources, clients must be able to lookup the host's name, but the amount of such traffic is small.

3.1 The LIN protocol

A location independent name is one that can map to different addresses over time. The LIN protocol manages these mappings, and involves three steps: setup, use, and tear-down. This discussion is aimed at those readers who are unfamiliar with DHCP [8], DNS [15, 16], and dynamic update extensions to these protocols [10, 23]. Readers familiar with the details of these protocols should

1. Metrocom maintains multiple coverage areas, and users who move between coverage areas, such as between San Francisco, CA and Washington D.C. must reestablish their connection and get a new IP address. Users who rely on transparent motion between coverage areas are not nomadic.
2. We use the term *travelling host* as a generic term to describe a computer moving between networks. Although previous work in this area used the term *mobile host*, we felt this term was too closely tied to the concept of Mobile IP and so added confusion to our discussion.

feel free to skim this section. We distinguish between the home network, corresponding to the traveller's main administrative domain, and the local network, the traveller's current locale.



This figure shows the basic steps in establishing a mapping between a travelling host's new location and its name. In Step 1, the travelling host requests a local address from the local network's DHCP server. In Step 2, the TH rebinds its name to this local address (ARR) using DNS's dynamic update protocol. In Step 3, the TH informs the local DHCP server of its name using a new option to the DHCP protocol. The DHCP server verifies the name using a standard authoritative DNS lookup in Step 4, and updates the local DNS server with the reverse name binding (PtrRR) in Step 5. At this point, the TH can communicate just as any other host on the local network.

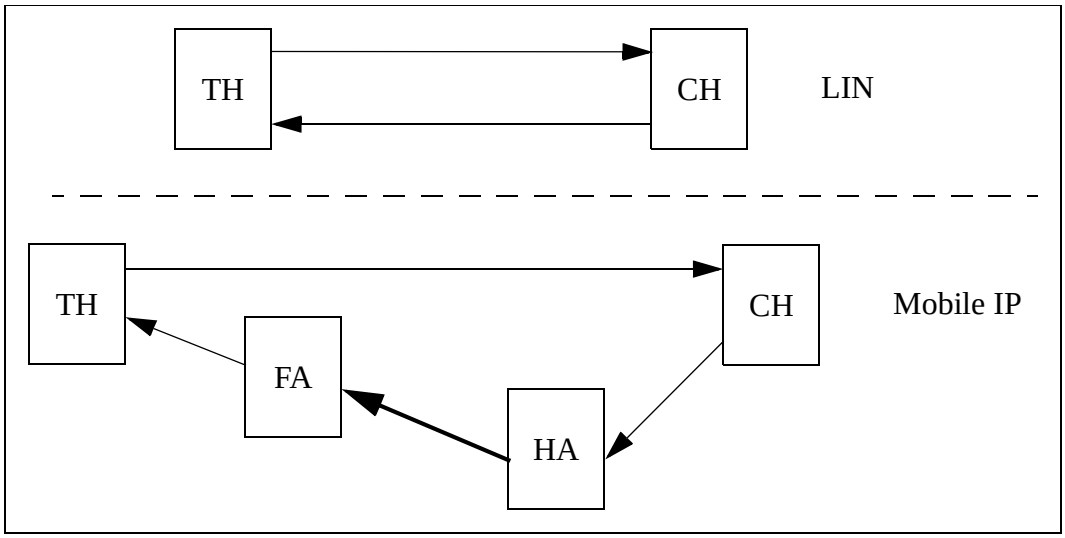
Figure 1: Binding names to new addresses using LIN

Figure 1 lists the steps to setup a LIN name-to-address binding. The process begins when a travelling host requests an address on its local network using DHCP [8]. The DHCP server responds with an otherwise unused IP address for this network and a *lease time*. The lease time specifies the period of time for which the travelling host can use this address. If the travelling host wishes to employ the address for longer than the lease, it can extend the lease by contacting the DHCP server.

Given the new IP address, the travelling host sends a message to its home DNS server using the DNS dynamic update protocol [10]. The message contains a request to delete the travelling host's last name-to-address binding (known as the "A resource record", or ARR) and a request to add an ARR with the new IP address and the length of time for which the address will be valid. This length is the minimum of the DHCP lease time and the travelling host's estimate of how long it will stay at this address. This estimate may be inferred from history or supplied by the travelling host's user. Upon receipt of this message, the home DNS server updates the corresponding entry to reflect the new address, and sets the time-to-live (TTL) for this entry to be the minimum of the time supplied by the travelling host and a configurable maximum TTL time. For reasons of security, a travelling host is allowed to rebind only its name and must authenticate with the home DNS in order to do so [9].

Upon receipt of acknowledgment from the home DNS server, the travelling host informs the local DHCP server of its name using a new option to DHCP, which we discuss in Section 4.9. The DHCP server first verifies that the name is correct by looking up the name-to-address binding from the travelling host's home DNS server using an authoritative DNS lookup. If the returned IP address matches the one it gave to the travelling host and the TTL on the DNS entry is less than or equal to the time remaining on the address's lease, then the DHCP server can trust the name is valid. Note that this verification does not weaken or extend the normal trust relationships of the Internet, since the home DNS server is the authority for name-to-address bindings for its domain.

Upon validation of the name, the local DHCP server rebinds the IP address to the travelling host's name at the local DNS server using the DNS dynamic update protocol. The time-to-live for this address-to-name binding (PTR resource record, or PtrRR) is set to match the TTL returned from the home DNS server as part of the name verification. This requires a trust relationship between the DHCP server and the local DNS server, which is reasonable since both are maintained by the same administrative staff [23].



This figure shows the steps to communicate using LIN and Mobile IP (as represented by the IETF IPv4 standard [19]). With LIN, the travelling host (TH) and the correspondent host (CH) communicate directly, since the nomadic host is just like any other Internet host while it is stationary. With Mobile IP, the travelling host can send packets directly to the CH, but return packets are routed to the home network by the Internet routers. A home agent (HA) on the home network transmits packets to the foreign agent (FA) on the TH's current network using an IP tunnel, which is represented in the figure as a darker arrow. The FA forwards the messages to the TH.

Figure 2: Correspondence using LIN vs. Mobile IP

At this point, the travelling host can communicate with any other Internet host as if the local network were its home network, and other hosts can communicate with it using its hostname. Figure 2 shows the steps needed to communicate between the travelling host and a correspondent for both LIN and Mobile IP. With LIN, the travelling host can initiate connections with any host using

its local IP address as the return address. The correspondent host's replies are sent directly back to the travelling host's local network since the source address in its packets correctly identify its location. Other hosts can initiate communication with the travelling host by looking up its name. As in normal operation, these lookups are sent to the travelling host's home DNS server, which returns the travelling host's local (current) IP address. Reverse name lookup requests get forwarded to the local DNS server, which satisfies them using the correct name-to-address mapping. The travelling host can renew the TTL on its mappings using steps 2 through 5 of the LIN protocol (in Figure 1) independent of renewing its lease, since the TTL can be shorter than the lease time.

There are two ways of destroying the name-to-address binding. The first method, and the one we advocate, is to wait for the TTL to elapse, at which point both DNS servers will flush the mapping from their databases. The second method is to inform the DHCP and DNS servers to revoke the mapping when the travelling host has moved. Unfortunately, this method is not sufficient to guarantee that the travelling host can move to a new address or that the DHCP server can reuse the travelling host's address since other sites on the Internet may have cached the name-to-address mapping. This issue is discussed in detail in the 4.2.

3.2 Trust Relationships in LIN

One of the advantages of LIN is that it uses existing trust relationships. LIN uses the home DNS server to satisfy name lookup requests for the travelling host's name, since it has authority for name-to-address mappings (ARRs) for names in its domain. Similarly, LIN uses the local DNS server to satisfy reverse name lookups, since the local DNS server is the authority on PtrRRs for the local subnetworks. Currently, these two servers are typically within the same administrative domain since there is a one-to-one-to-one mapping between name, IP address, and network location in the absence of mobility. With LIN, these servers can be in different domains without violating the correctness of DNS or introducing security holes

With LIN, the local DHCP server must trust the DNS system, and the home DNS server in particular, to correctly resolve the travelling host's name. As a result, attacks on the DNS system could present problems for LIN. However, the DNS system is core to the proper functioning of the Internet, and steps are being taken to ensure the integrity of DNS [12]. LIN does not conflict with these proposals.

The weakest point in the security chain, the travelling host, is only granted limited trust. The home DNS server trusts the travelling host to provide a correct IP addresses for itself. A malicious or faulty travelling host could supply an IP address for some other network, but the effect of this record is equivalent to a misconfigured DNS server and mechanisms already exist to handle this situation.

A more subtle point is the trust relationship between the travelling host and local network. The travelling host must trust the local network to correctly deliver its packets, to give it a valid IP address, and to perform reverse-name lookup correctly. This is true with or without LIN. The local network has to trust the travelling host enough to give it an IP address and access to its network, but no more. The host name supplied by the travelling host in step 3 of LIN is purely informational, the DHCP verifies the name before using it. If the name lookup fails, if it returns the

wrong IP address, or if the TTL is longer than the lease time, the DHCP server can raise a security exception and refuse to grant further access to the travelling host. Most importantly, the local network does not need to provide special privileges to the travelling host, such as the ability to tunnel through its firewall. As a result, networks with firewalls that allow only connections using blessed protocols (such as DNS lookup and update but not IP tunnels) can allow visiting travelling hosts via LIN but not via Mobile IP.

4 Technical Details

This section discusses the technical details and implications raised by the LIN protocol.

4.1 How does the DHCP know to trust the travelling host?

In order to safely grant an IP address to the travelling host, the DHCP server must first trust the travelling host to use the local network. DHCP supports two forms of trust: trust anyone on the local network or trust only those machines using a known media-access control (MAC) address. The former alternative is typically used in situations where security is not an issue, such as universities, or where users must supply a password in order to use a network port. In the latter alternative, the travelling host's user must request access to the network by supplying her credentials and the machine's MAC address. In either alternative, the local network administrator either knows the identity of the travelling host or implicitly trusts them.

Because LIN uses DHCP to rebind its name, the travelling host has greater accountability than it would without LIN. For example, Web server administrators use reverse name lookup to determine the source of requests. With LIN, the reverse name lookup returns the travelling host's real name. If the TTL has already expired when the reverse lookup occurs, the DHCP server can examine its logs to determine who has been using the IP address.

4.2 Stale DNS cache entries

For reasons of scalability, the DNS system caches name-to-address bindings at various locations throughout the Internet as a side effect of name lookup. LIN dynamically changes the mapping between a name and its address, and the likelihood of stale cache entries grows with the rate of movement of the travelling host. Unfortunately, DNS servers do not currently track where their resource records may be cached and so have no way of directly revoking cached entries. In addition, name lookups that are satisfied by cache entries are never seen by the home DNS server, so it cannot detect and flush stale entries on use. These stale entries will cause packets meant for the travelling host to be sent to the wrong address, resulting in transient communication failure until the TTL for the stale cache entry expires.

LIN avoids the problem of stale cache entries by using temporary updates with conservative TTLs that underestimate the length of time the travelling host will remain at its current location. As a result, both the database and cache entries will be flushed before the IP address contained in the entry becomes invalid. As described in Section 3.1, LIN selects a TTL that is the minimum of the DHCP lease time, the travelling host's estimate of its length of stay, and a threshold value. The threshold is enforced by the home DNS server to prevent denial of service attacks on the DHCP from the travelling host, and can be set to some reasonable value such as an hour.

However, once an entry has been cached there is no way to update its TTL or flush the entry before its TTL expires. Although the travelling host can reconnect to a new network before its old mapping's TTL has expired, some Internet hosts may be unable to connect to the travelling host due to stale DNS cache entries. Fortunately, nomadic users do not typically move between networks quickly. For instance, typical commutes range from tens of minutes to over an hour, and one must wait for the airplane to reach a sufficient altitude before using a laptop. Using TTLs that are smaller than the time to move to a new location avoids the problem of stale cache entries.

4.3 The effect of short TTLs on DNS load

LIN's use of short TTLs to prevent stale cache entries has the drawback of lowering the effectiveness of caching for that entry. As a result, LIN may increase the number of requests a DNS server must handle, increasing its response time and the load on the home network. However, the increase should be insignificant as long as the travelling host's name is not frequently accessed. Fortunately, nomadic hosts tend not to be busy internet servers since they suffer periods of disconnection, are more likely to be stolen or lost, and are difficult to backup and maintain. In addition, the increase in load due to name lookup is likely to be smaller than the load due to triangular routing and encapsulation. Name lookups happen at most once per TTL per incoming connection, whereas the triangular routing and encapsulation needed by MobileIP affects the transmission of each packet sent to the mobile host.

The relationship between TTLs, DNS load, and mobility provides an interesting dynamic tradeoff. Travelling hosts can arbitrarily lower their TTLs if they expect to move shortly, use low TTLs if they expect to move frequently, or request longer TTLs if they expect to remain stationary for some time. At one extreme, the TTL is set to zero and the travelling host's entry will not be cached. This allows the host to move between networks spontaneously, as for instance it would if using a digital cellular modem across provider coverage areas. At the other extreme, the TTL is set to equal the home DNS server's threshold. As long as the threshold is reasonably high, such as an hour, the load will be trivial. In general, we expect the travelling host to use the lease provided by the DHCP server since most nomadic users are unaware of their network settings and tend to move between networks slowly. However, aggressive users or advanced operating systems may manipulate this value to provide greater mobility with small likelihood of disconnection due to stale cache entries.

4.4 How can the travelling host securely update its home DNS entry?

In general, allowing arbitrary hosts to securely update DNS entries is a difficult problem. Fortunately, LIN only requires that a host be able to update its own entry. Since the host is known to the DNS server, it can establish a shared secret with the server before travelling. The server binds the secret to the entry, and authenticates any update message using the secret. The secret could be a public or private key. For example, the DNS secure update protocol [11] or any secure variant of the Needham-Schroeder protocol [18] would suffice.

We propose to use the TSIG protocol [9] for this purpose. TSIG is currently being developed for inclusion in DNS. TSIG uses a shared secret between a host and its DNS server for authentication, which means that a travelling host can securely update its name-to-address binding regardless of its location. Since TSIG is still a proposed extension to DNS, we have approximated the TSIG protocol using the secure shell (SSH) protocol [28]. SSH is a secure version of rsh that allows a

host to execute commands on a remote site securely. In our test environment, the travelling host executes a program on the home DNS server that performs the client portion of the dynamic update protocol. The input to the program is the host's new IP address and requested TTL, the program returns the actual time used by the DNS server.

4.5 What happens if the travelling host moves to a new location before its lease expires?

If the travelling host moves to a new network with a new IP address, its home DNS server will be updated and new connections that reach the DNS server will see the travelling host's new address. In addition, the travelling host will be able to initiate connections to other Internet hosts normally. However, it is likely that there will be stale cache entries whose lease on life has not yet expired, and these stale entries may prevent others from initiating communication with the travelling host. This problem can be avoided if the travelling host can anticipate its movement by supplying a short lease time to the home DNS server in step 2 of LIN. If this issue proves to be a problem, LIN could be extended to solve this problem. Section 5 presents one alternative.

4.6 When can the local DHCP server reuse the travelling host's IP address?

The DHCP protocol allows a visiting host to release its IP address before the lease expires, thus freeing the IP address for reuse by another host. This reuse poses a problem for LIN, since the binding of that IP address to the previous host's name may still exist in caches at various DNS servers throughout the Internet. As a result, the host assigned this new IP address may get packets intended for the previous lessee of that address. The safe course, and the one we adopt in LIN, is not to pre-release the IP address.

This does raise the problem of a misbehaving travelling host that releases its address early in order that packets addressed to it will instead be delivered to some unsuspecting future client of the DHCP server. However, this is not a serious new attack since it is already possible to address packets to any IP address. If it does prove to be a problem, the DHCP server could ignore the pre-release of IP addresses assigned via LIN (detectable by the use of the new option discussed in Section 4.9) until the TTL has expired. If the travelling host requests another address from this server, it would be given the same address it previously held.

4.7 What changes to the travelling host are needed for LIN?

In LIN, the travelling host orchestrates the various steps, and as such requires most of the changes for LIN. First, the travelling host must be configured to use DHCP, to refer to its home DNS server, and to have a well-known name. Next, the DHCP client on the travelling host must be extended to perform the various steps of LIN. In particular, it must act as a DNS dynamic update requestor [10], must inform DHCP of its fully qualified name, and must be able to renew its lease and the TTL on its ARR and PtrRR periodically.

4.8 What changes to DNS are needed for LIN?

LIN requires two additions to DNS (as of BIND version 8.1 [4]): the addition of a maximum TTL for travelling hosts, and the addition of temporary ARRs and PtrRRs. The former is needed to prevent a malicious travelling host from using a TTL that is longer than the DHCP's lease time, and thus maintaining a bogus mapping for its address. The latter is needed to ensure that the LIN binding is destroyed automatically at all the home's primary and secondary DNS servers. The use of temporary RRs is practical independent of LIN, for example it would ease the task of revoking a

mapping of a host when it is moved by avoiding the complications induced by zone transfers [1]. Both additions require the DNS server be able to identify LIN requests.

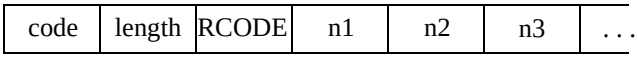
We propose two mechanisms that can achieve this effect. The first, and the one we advocate, is to add a flag to the DNS message header that marks the record as temporary. The (authoritative) DNS server would treat this update as a regular ARR or PtrRR, but would mark it as temporary so that its TTL would tick down in the same manner as a cache entry. As a result, the TTL it would use for query responses would become smaller over time. When the TTL expired, the DNS server could either flush the entry or preserve it for bookkeeping until it was explicitly deleted in the next update from the client. In this case, the DHCP server should expect the temporary flag to be set when it receives the authoritative answer. Other DNS requestors could safely ignore the flag.

The second mechanism would use negative TTLs in update requests to imply the temporary status. This is similar to a proposal contained in the original DNS implementation RFC for representing absolute vs. relative times in the DNS database [16]. However, there would be no way for a DHCP server to ensure that a travelling host's ARR was temporary. In either approach, only the primary DNS server for the home zone and its secondaries need to support these change. Section 5.3 describes an alternate approach to modifying DNS.

Two subtle points in the interaction between LIN and DNS need to be discussed. First, the TTL used in the ARR may be different than that requested by the travelling host, but the protocol does not require that the RR is returned in the response and does not specify if it will contain the requested RR or the one inserted in the database. Extending DNS in either of the two ways mentioned above would allow us to ensure that the new TTL is returned to the client. Second, the DNS specification contains a minimum TTL in the *start of zone authority (*SOA) RR that is used to provide a minimum TTL on all responses [16]. Fortunately for LIN, most sites do not implement this feature [2]. For sites that do enforce a minimum, the administrator could have a separate DNS server for travelling hosts with a minimum of zero. Since travelling hosts are nomadic, this server should not suffer excessive loads as a result of the lower minimum.

4.9 What changes to the local DHCP server are needed for LIN?

The only modification to DHCP is the addition of a new option, shown in Figure 3. This option allows the travelling host to inform the DHCP server of its fully qualified domain name to enable reverse name lookups to return the travelling host's name. Use of this option also informs the DHCP server that this host is using the LIN protocol. The option is sent as part of a lease renewal request; upon receipt of this option, the DHCP server first validates the address in its database and then validates the name via an authoritative DNS lookup of the name. This could be done, for example, by performing an SOA query to determine the primary master for the zone, and then contacting that server directly to lookup the travelling host's ARR authoritatively. A beneficial side effect of using the SOA query is that it returns the email address of an administrator in that domain who can be contacted if the travelling host misbehaves! Upon receipt of the ARR, the DHCP server formulates a PtrRR using the IP address's IN-ADDR.ARPA domain name and the TTL contained in the ARR. As part of the DNS dynamic update, it should remove any previous PtrRRs for this address from the local DNS server.



This diagram shows the format for the new option to DHCP to specify the travelling host's name. Length is the number of characters, RCODE is the return code from the local DNS server update, and n1, n2, ... are the first, second, etc characters of the name (up to 128 characters).

Figure 3: Format of DHCP option for step 3 of LIN

Figure 3 shows the format for the new option record we are proposing. It is essentially the same as the proposed option 81 [23], but serves to notify the DHCP server that this host is using the LIN protocol. This notification is needed to add LIN-specific behavior to the DHCP server, in particular the validation of the name via an authoritative lookup to the home DNS server and special treatment of leases belonging to LIN clients. LIN could be implemented without this option, but doing so would require some additional trust relationships to be established, for instance the local DHCP server would have to trust the name given to it by the travelling host.

Because the DHCP server uses the ARR from the home DNS server as the basis for its dynamic update of the local DNS server, the two entries will have identical names, addresses, and TTLs even though they are now stored in separate security domains. Although it is not strictly necessary that ARR and PtrRR agree, it should reduce any confusion that may result.

5 Extensions to LIN

Currently, LIN has three drawbacks: it does not provide the transparent mobility of Mobile IP; hosts must wait until their TTL has expired before connecting to a new network or risk communication failure; and it requires changes to DNS and DHCP. This section discusses extensions to our protocol that address these shortcomings. However, these extensions are merely proposals and require additional support from the network infrastructure. As such we leave them as future work.

5.1 Adding transparent mobility

The first drawback involves the lack of transparent mobility in LIN. There are two approaches to solving this problem. The first is to add a level of indirection at the IP layer in the correspondent host. The travelling host notifies all current correspondents of its new location. An IP level extension on that host handles these packets, and caches an IP address to IP address mapping between the old location (from which the connection was originally established) and the new one. If the correspondent host receives a new location for a host with which it communicates, it replaces the old mapping with a new one. This mechanism is very similar to ones proposed for MobileIP Route optimization (e.g., [17, 20, 22]). Alternatively, one could add support to higher level protocols that are layered on IP. For instance, TCP could have a control packet that requested the receiver to update the IP address contained in the connection state to the new address. Since the travelling host has connection state for every open connection, it would be an easy matter to create and send these messages to all of its correspondents.

5.2 Reducing the minimum time to move between networks

Another drawback is that correspondents to a travelling host that moves to a new location before the TTLs on the existing address expire may encounter communication failures due to stale

ARRs. Although the travelling host can limit this problem by specifying small TTLs, it may not be able to anticipate its movement or the time it will next connect from a different location.

One solution is for the travelling host to leave an agent at its old location that will forward packets to the client at its new location. This agent is very similar to the home agent of MobileIP, but need only exist from the point a client connects to a new network until the old address's lease on life expires. Since this is likely to be a small period of time (certainly less than an hour), the additional network load of redirecting packets should be small. For instance, the agent could be implemented as an active network filter on a smart router [27].

5.3 Alternatives to modifying DNS and DHCP

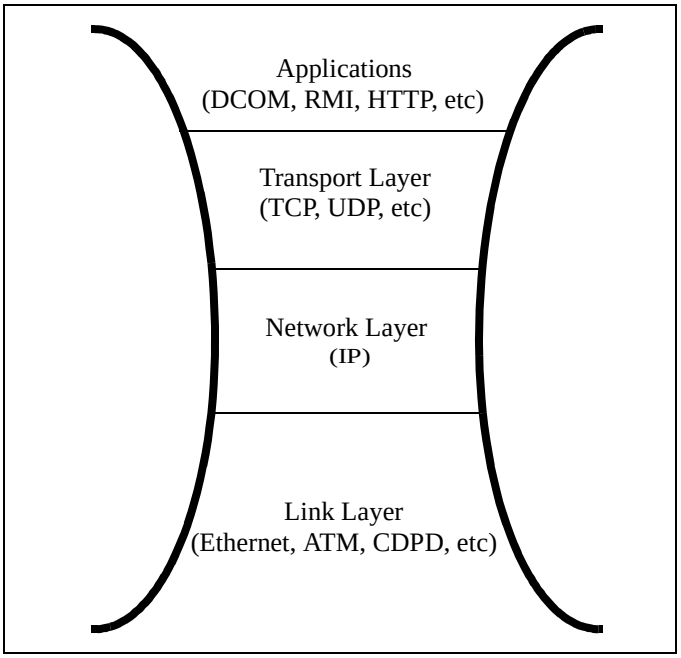
The third drawback of LIN is the need to modify DNS and DHCP. We could avoid modifying the DHCP server if the network administrator trusted the travelling host to supply the correct name and to remove the ARR from its home DNS server when its lease expired. Similarly, we could avoid modifying DNS by setting the TTLs for ARRs and PtrRRs to zero, but it is still incumbent on the travelling host to remove the ARR containing the local address from its home DNS server. Failure to do so is a denial-of-service attack, since the local DHCP server cannot safely reuse that IP address until the ARRs mapping is deleted.

One way to counter this form of attack is to leave the lease record active until the home DNS server deletes the corresponding ARR. If the travelling host attempts to get another IP address, the DHCP server will send it the old address instead (and possibly raise a security exception to the administrator). If not, the DHCP server can periodically perform a DNS lookup on the errant travelling host's name to see if the ARR has been removed. When it has, the DHCP server can safely revoke the lease. If a sufficiently long period has elapsed without the ARR being removed, the DHCP can send e-mail to the administrator of the home DNS server requesting that the entry is manually removed.

6 Related Work

There are effectively four places to support mobility: at the media layer, the network layer, the transport layer, or the application layer. You can view these different layers as part of an hourglass, shown in Figure 4. Because the network layer is the smallest part of the architecture, it is a natural place to which to add support for mobility. Adding mobility to other layers requires more work because there are a plethora of protocols that must be modified. For example, supporting mobility at the transport layer would require modifying all implementations of TCP, UDP, and any other protocol layered on top of raw IP. Since we are interested in supporting mobility over the Internet, we are mainly concerned with solutions that work for all major transport protocols, applications, and media types and hence focus on related work supporting mobility at the network layer.

There is a variety of work that has tried to support mobility at the network layer, Tasman presents a good summary [26]. Of particular relevance to LIN are those methods that attempt to work with existing Internet infrastructure, since they are most easily deployed. For brevity, we focus the discussion on these. Section 6.1 describes Mobile IP in more detail, and Section 6.2 discusses approaches that use DNS and DHCP.



This figure shows the “hourglass” architectural model of the Internet. The waist of the hourglass is the ideal place to support mobility since it involves the least amount of change. The waist corresponds to the network level that consists of the IP protocol, which explains why most mobility research is focused at this level.

Figure 4: The hourglass architectural model of the Internet Protocol

6.1 Comparison to Mobile IP

Mobile IP is an approach to supporting mobile computers that allows a computer to keep both its name and IP address while travelling. Since higher protocols use the IP address as part of the connection state, maintaining the IP address allows hosts to move transparently between networks. Unfortunately, maintaining the IP address over several locations breaks the implicit one-to-one mapping between address and location, which leads to several drawbacks.

Traditionally, a host’s IP address uniquely identifies the network to which the host is connected. The Internet uses this binding to route packets destined for a particular host to its network. In Mobile IP, a mobile host (MH) keeps the same IP address as it moves from network to network. This breaks the assumptions made by standard routing mechanisms since the IP address no longer indicates the network to which to send packets destined for the MH. Different Mobile IP protocols propose different solutions to this problem, but all involve special support to redirect packets and/or IP encapsulation. These mechanisms result in performance degradation and conflict with existing Internet security mechanisms.

Redirection of packets can happen in one of several ways. The current IPv4 IETF standard requires an agent on the MH’s home network [19]. This *home agent* listens for any packets destined for the MH and retransmits them to the mobile host by encapsulating them in an IP tunnel. The other end of the tunnel is either maintained by the MH itself or a *foreign agent* on the MH’s current network. This redirection is known as triangle routing, since 2-way communication

between a MH and its correspondent host (CH) must travel through the home network. This results in additional load on the network, requires special servers to exist at the home network, and requires the use of IP tunneling. Unfortunately, the case where the MH and the CH are co-located is both the worst case and a common scenario.

The chief solution to triangle routing is to inform the CH of the MH's new location and have it use the MH's new address instead. However, this requires updating the IP layer of the corresponding host's operating system. In particular this means that the MH cannot communicate with any existing Internet host, except the handful of research machines that support the correct Mobile IP protocol [17], or machines that support the IPv6 IP proxy mechanism [22].

To route packets to the MH using its permanent IP address, the CH or the home agent encapsulates packets destined for the MH by adding IP headers that contain the MH's real address. This encapsulation has several drawbacks. First, many network administrators consider this form of encapsulation to be a security breach, as it bypasses standard firewall mechanisms. Hence many security conscious administrators are configuring their firewalls to detect and abort IP tunnels [5].

Second, the extra header may make a packet larger than the maximum transmission size, forcing the packet to be fragmented. This can break transport layer flow control mechanisms resulting in a higher load on the network and a loss of efficiency, which is especially problematic for connections with real-time needs. Zhao and Baker determined the performance penalty of encapsulation and triangle routing can be substantial: doubling latency and reducing bandwidth by 45% [29]. Perkins [21] and Johnson [13] both proposed a similar technique based on loose source record routing that suffered even higher performance penalties than IP encapsulation [14].

A third problem with Mobile IP comes from source address filters. Network administrators use these to prevent IP spoof attacks in which a malicious host generates spurious source IP addresses to hide the source of the attack. Source address filters prevent this spoofing by destroying packets whose source address does not correspond with the network from which they originate. Unfortunately packets from a MH will also be filtered, since the MH's IP address does not belong to the MH's current network. The only way to bypass these filters is to encapsulate the packets by tunneling them either to the home agent or the correspondent host, incurring even higher performance penalties [29].

Chesire and Baker present a solution to these problems by allowing the MH to choose the optimal communication strategy based on several factors [7]. Unfortunately, their approach suffers the same basic limitations as MobileIP: transparent mobility either requires special support in the correspondent host or breaks the one-to-one correspondence between address and location.

6.2 Supporting mobility with DNS and DHCP

The other alternative is to sacrifice transparency in order to avoid the problems associated with Mobile IP. This is the approach we advocate for nomadic computing, and can be achieved with three degrees of support. The lowest level of support, represented by DHCP [8], provides an IP address for the local network, but does not map that address to the travelling host's well-known name. As a result, travelling hosts can initiate communication with other hosts on the network, but other hosts cannot directly access resources on the travelling host without out-of-band communi-

cation. For instance, a correspondent host could not open an FTP connection to the TH with DHCP alone.

The next level of support is provided by DHCP with dynamic DNS update [10, 23]. This allows the client or the DHCP server to add or delete ARRs or PtrRRs at the DNS server. Using option 81, the client can either tell the DHCP server to add the appropriate ARR or assert that it will make the update itself. In either case the DHCP server must add the appropriate PtrRR. This approach will provide the benefits of mobility within a security domain. For example, Brian Dols and Lans Carstensen at Rose-Hullman Institute of Technology [6] have implemented these protocols, allowing student to connect their laptops to any subnetworks on their campus without reconfiguring. These laptops maintain the same name as they move on campus, but get addresses corresponding to the network to which they are currently connected. However, we are the first to suggest this same mechanism can be easily extended (with temporary DNS RRs and additional support at the DHCP server) to allow movement between security domains. We believe this is a significant advantage for nomadic users, particularly those who travel widely.

7 Conclusion

LIN extends DNS and DHCP to support nomadic computing without requiring additional infrastructure support or modification to the existing trust relationships on the Internet. LIN utilizes existing and proposed Internet standards with only minor extensions, so travelling hosts can communicate without suffering the performance and security drawbacks of MobileIP. In addition, with LIN a correspondent host can communicate with a travelling host transparently, that is without requiring any knowledge that its correspondence is directed to a portable computer. LIN gets these advantages by violating the transparency of mobility, since the correspondent host knows the travelling host's actual address. However, this tradeoff is reasonable for nomadic computers that do not need to maintain connections between networks.

We view LIN as a "lowering of the bar," enabling network administrators to deploy support for mobile users with less effort. We are working towards having our extensions included in standard releases of DNS and DHCP, which will enable widespread use of LIN. In the meantime, individual sites can upgrade their DNS and DHCP servers to both support their own and visiting nomadic users seamlessly. Once nomadic computing is widespread, network administrators may be more motivated to add support for true mobility ala MobileIP.

Bibliography

[1] P. Albitz and C. Liu. *DNS and BIND*, page 167. O'Reilly and Associates, Inc., second edition, January 1997.

[2] Ibid., pages 166–167.

[3] B. J. Bennington and C. R. Bartel. Wireless Andrew: Experience building a high speed campus-wide wireless data network. In *Proceedings of the Third ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom '97)*, September 1997.

[4] BIND version 8.1. Internet Software Consortium. Available as http://www.isc.org/bind.html.

[5] J. Binkley. Security considerations for mobility and firewalls. Under preparation as an Internet Draft.

[6] Lans Carstensen, April 1997. Personal communication.

[7] S. Chesire and M. Baker. Internet mobility 4x4. In *Proceedings of SIGCOMM '96*, August 1996.

[8] R. Droms. Dynamic host configuration protocol. IETF Network Working Group Request for Comments (RFC) 2131, March 1997. replaces RFC 1541.

[9] P. Vixie (Ed.), O. Gudmundsson, and D. Eastlake III. Secret key transaction signatures for DNS (TSIG). IETF DNSIND Working Group Internet draft draft-ietf-dnsind-tsig-04.txt, March 1998. Amends RFC 1035, expires October 1998.

[10] P. Vixie (Ed.), S. Thomson, Y. Rekhter, and J. Bound. Dynamic updates in the domain name system (DNS UPDATE). IETF Network Working Group Request for Comments (RFC) 2136, April 1997. Updates RFC 1035.

[11] D. Eastlake III. Secure domain name system dynamic update. IETF Network Working Group Request for Comments (RFC) 2137, April 1997. Updates RFC 1035.

[12] D. Eastlake III and C. Kaufman. Domain name system security extensions. IETF Network Working Group Request for Comments (RFC) 2065, January 1997. Updates RFCs 1034, 1035.

[13] D. B. Johnson. Mobile host internetworking using IP loose source routing. Technical Report CMU-CS-93-128, School of Computer Science, Carnegie Mellon University, February 1993.

[14] D. B. Johnson and D. A. Maltz. Protocols for adaptive wireless and mobile networking. *IEEE Personal Communications*, 3(1):34–42, February 1996. Special issue on Mobile Computing at Carnegie Mellon.

[15] P. Mockapetris. Domain names - concepts and facilities. IETF Network Working Group Request for Comments (RFC) 1034, November 1987.

[16] P. Mockapetris. Domain names - implementation and specification. IETF Network Working Group Request for Comments (RFC) 1035, November 1987.

[17] A. Myles, D. B. Johnson, and C. Perkins. A mobile host protocol supporting route optimization and authentication. *IEEE Journal on Selected Areas in Communications*, 13(5):839–948, June 1995. Special issue on Mobile and Wireless Computing Networks.

[18] R. Needham and M. D. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21(12):993–998, December 1978.

[19] C. Perkins. IP mobility support. IETF Network Working Group Request for Comments (RFC) 2002, October 1996.

[20] C. Perkins and D. B. Johnson. Route optimization in mobile IP. IETF Mobile IP Working Group Internet Draft draft-ietf-mobileip-optim-07.txt, November 1997. Expires May 20, 1998.

[21] C. E. Perkins and P. Bhagwat. A mobile networking system based on internet protocol. *IEEE Personal Communication*, 1(1):32–41, 1st Qtr 1994.

[22] C. E. Perkins and D. B. Johnson. Mobility support in IPv6. In *Proceedings of the Second*

Annual International Conference on Mobile Computing and Networking (MobiCom '96), November 1996.

[23] Y. Rekhter. Interaction between DHCP and DNS. IETF Network Working Group Internet draft draft-ietf-dhc-dhcp-dns-07.txt, February 1998. expiration Aug 1998.

[24] M. Satyanarayanan. Fundamental challenges in mobile computing. In *Fifteenth ACM Symposium on Principles of Distributed Computing*, May 1996.

[25] Metrocom Inc support staff. personal email communications.

[26] M. P. Tasman. *Protocols and Caching Strategies in Support of Internetwork Mobility*. PhD thesis, University of Wisconsin - Madison, 1994.

[27] D. L. Tennenhouse, J. M. Smith, W. D. Sincoskie, D. J. Wetherall, and G. J. Minden. A survey of active network research. *IEEE Communications Magazine*, 35(1):80–86, January 1997.

[28] T. Ylonen, T. Kivinen, and M. Saarinen. SSH authentication protocol. IETF Network Working Group Internet Draft draft-ietf-secsh-userauth-03.txt, November 1997. Expires May 7, 1998.

[29] X. Zhao and M. G. Baker. Flexible connectivity management for mobile hosts. Technical Report CSL-TR-97-735, Computer Systems Laboratory, Departments of Electrical Engineering and Computer Science, Stanford University, September 1997.