

Universal Mobile Addressing

Jorge A. Cobb
Chris C. Edmondson-Yurkanan
Mohamed G. Gouda

Department of Computer Sciences
University of Texas at Austin
Austin, Texas 78712-1188

Abstract

To support mobility, we propose to use a triple (M, N, d) as an address for every stationary or mobile computer in a multi-network system, like the Internet. In this (M, N, d) address, M is the network where the computer is currently located, N is the network where the computer is usually located, and d is a unique identifier of the computer in network N . The address of a stationary computer is always (N, N, d) , while the address of a mobile computer is (M, N, d) , where M may be different from N . In this paper, we discuss how to use this universal addressing in a number of protocols for routing data messages to mobile computers, for establishing and maintaining mobile virtual circuits, and for supporting mobile groups.

1. Introduction

We consider a system, like the Internet [2] and [6], that consists of several computer networks. The computers in each network are classified into hosts and routers. Each network has one or more hosts and one or more routers. For convenience, we assume that each host is attached to exactly one network, and each router is attached to two or more networks.

Hosts generate and send data messages destined to other hosts, possibly in other networks, in the system. Hosts also receive and consume data messages that are destined to them. The function of routers is to forward the sent data messages from one network to an-

other until each message arrives at its destination host.

Each host has a unique name of the form (N, d) , where N is the network to which the host is attached, and d is a unique identifier for the host in network N .

Each router has a set of two or more unique names of the form

$$\{ (N_0, d_0), \dots, (N_r, d_r) \}$$

where N_0, \dots, N_r are the networks to which the router is attached, and every d_k is the unique identifier of the router in network N_k .

When a host generates a data message, it augments the message with the name of its destination host. Thus, each data message is of the form

$$\text{data}(N, d, \text{text})$$

where (N, d) is the name of the destination host.

When a router receives a $\text{data}(N, d, \text{text})$ message from some attached network M , the router uses N to compute another attached network K , and forwards the message to another router in network K , and the cycle repeats. This cycle continues until finally some router, attached to the destination network N , receives the message and forwards it to the destination host (N, d) .

These naming and routing schemes are adequate as long as hosts are "stationary", i. e. remain attached to their networks indefinitely. However, other schemes are needed to ac-

commodate "mobile" hosts that can move between different networks in the system. Existing schemes for accommodating mobile hosts are surveyed in [4], and some schemes to solve the problem in local-area or campus networks are proposed in [1] and [3].

One scheme for accommodating mobile hosts in wide-area networks is suggested in [5] and [7]. In this scheme, when a mobile host (N, d) moves to a network M , it is assigned a temporary name (M, c) by some agent in network M . In this case, any data message destined to host (N, d) is eventually re-routed to host (M, c) . When host (N, d) leaves network M and moves to network K , it is assigned another temporary name (K, b) by some naming agent in network K . The temporary name (M, c) is returned to the naming agent in network M so that it can be assigned to other mobile hosts that move to network M .

The problem of this scheme is that it requires a naming agent in each network in the system. The function of these agents is to keep track of the temporary names in their networks.

In this paper, we discuss another scheme to accommodate mobile hosts. This scheme does not require naming agents for assigning and reclaiming temporary addresses in any network in the system. An informal presentation of this scheme is given in Section 2. Then, a detailed and formal presentation of the scheme is given in Sections 3 through 6. The impact of mobility and the use of our addressing scheme on virtual circuits and mobile groups is discussed in Sections 7 and 8. Section 9 describes the relationship of our scheme to current IP addressing. We conclude the paper with some final remarks in Section 10.

2. Mobile Addresses

In a system with only stationary computers, the pair (N, d) serves both as a name and as an address. That is, because (N, d) is unique for each computer, it can be viewed as a name, and since it indicates that the computer is in network N , it can be viewed as an address.

For a system with mobile hosts, the pair (N, d) can no longer be viewed as an address, since the host need not always be located in network N . Thus, we keep (N, d) as a name and develop a new addressing scheme.

Our scheme for addressing mobile hosts is as follows. When a host (N, d) moves to a network M , the host assumes the address (M, N, d) . Host (N, d) continues to assume this address as long as it is attached to network M . When host (N, d) finally leaves network M for another network K , it assumes the address (K, N, d) , and so on. Note that because the names of different hosts are distinct, and because host names are included in host addresses, the addresses of different hosts are distinct.

Note that a stationary host (N, d) can be viewed as a mobile host that never exercises its choice of moving to a network other than N . Hence, the address of this host is always (N, N, d) . In this manner, our addressing scheme is uniform for both mobile and stationary hosts.

Using this scheme, a data message that is sent to host (N, d) becomes of the form
$$\text{data}(M, N, d, \text{text})$$

where M is the current network of the destination host (N, d) .

When a host (N, d) moves to a network M and assumes the address (M, N, d) , it sends an $\text{upd}(M, N, d)$ message to each router in network N to inform the router that host (N, d) is currently attached to network M . Later, when another host sends a $\text{data}(N, N, d, \text{text})$ message destined to host (N, d) , the message ends up in a router in network N . This router recognizes that host (N, d) is currently in network M . Thus, it modifies the message, making it of the form $\text{data}(M, N, d, \text{text})$, and forwards the modified message towards network M . When the data message arrives at a router in network M , the router delivers the message directly to host (N, d) .

It is possible that by the time the $\text{data}(M, N, d, \text{text})$ message arrives at a router in network M , host (N, d) has already left network M . In this case, the message is discarded.

Before we present these addressing and routing protocols in more detail, we need to discuss how one computer, host or router, can send a message to another computer, host or router, in the same network. This is done next.

Each computer, host or router, in each network in our system has a unique hardware address in the range $1..n-1$. Henceforth, we adopt i and j to denote hardware addresses in our system, and refer to a host by either its hardware address i or its name (N, d) .

For a computer i to send a message $\text{msg}(f_0, \dots, f_r)$ to a computer j , where both i and j are in the same network M , computer i executes the command:

send $\text{msg}(j, f_0, \dots, f_r)$ **over** M

Thus, computer i needs to know beforehand the hardware address j of computer j .

Computer i can compute the hardware address of computer j from the name (N, d) of computer j by executing the command:

$\text{adr} := \text{arp}.(M, N, d)$

In this command, arp is a function that takes as inputs a network M and a computer name (N, d) and outputs either the hardware address of computer (N, d) if this computer is currently attached to network M , or 0 if this computer is currently not attached to network M .

For a computer i to send a message $\text{msg}(f_0, \dots, f_r)$ to every computer in its network M , computer i executes the command:

send $\text{msg}(0, f_0, \dots, f_r)$ **over** M

We are now ready to define our addressing and routing protocols in detail. The definitions proceed in three steps. First, a protocol for generating mobile addresses is given in Section 3. Second, a protocol for propagating mobile addresses is discussed in Section 4. Third, a protocol for routing data messages using the propagated mobile addresses is defined in Section 5.

3. Generation of Mobile Addresses

Periodically, each router in the system broadcasts its names using $\text{here}(N, d)$ messages, where (N, d) is a name of the router. When a mobile host i receives a $\text{here}(N, d)$

message from a nearby router, it stores N and d in two local variables CN and cr , respectively. Variable CN along with the name of host i constitute the current address of i . As shown below, variables CN and cr are also used by host i to send messages to the nearest router in the current network. The code for mobile host i is as follows.

var

CN : the network to which mobile host i is currently attached

cr : id of a router in network CN

action

rcv $\text{here}(0, N, d) \rightarrow CN, cr := N, d$

end

A router j periodically broadcasts its names. Each name (N, d) of router j is broadcasted over network N . The code for router j is as follows.

const

SM : the set of names of router j

action

timeout every τ msec \rightarrow

for every $(N, d) \in SM$ **do**

send $\text{here}(0, N, d)$ **over** N

rof

end

4. Propagation of Mobile Addresses

When a mobile host i receives a $\text{here}(N, d)$ message and detects that $CN \neq N$, which implies that host i has moved to a new network, host i sends an $\text{upd}(N, PN, pd)$ message, where (PN, pd) is the name of host i . Hence, the above action of mobile host i needs to be modified to become as follows.

const

PN : the network in the name of mobile host i

pd : the id in the name of mobile host i

```

action
  rcv here(0, N, d) →
    if CN ≠ N
    then
      CN, cr := N, d;
      adr := arp.(CN, CN, cr);
      send upd(adr, CN, PN, pd) over CN
    fi
end

```

When router j receives an $\text{upd}(M, N, d)$ message, it checks whether it is attached to network N . If router j is attached to network N , then it broadcasts an $\text{upd}(M, N, d)$ message to every other router in network N . If router j is not attached to network N , then it forwards an $\text{upd}(M, N, d)$ message to some router in an attached network K which is the next hop to network N . The name of this router is obtained by performing the function call route.N . Thus, the following two actions need to be added to the code of router j .

```

const
  SN : the set of networks to which
       router  $j$  is attached

var
  cnet: array [network, id] of network

action
  rcv upd( $j, M, N, d$ ) →
    if  $N \in \text{SN}$ 
    then
      cnet[ $N, d$ ] :=  $M$ ;
      send upd(0,  $M, N, d$ ) over  $N$ 
    else
      ( $K, c$ ) :=  $\text{route.N}$ ;
      adr := arp.( $K, K, c$ );
      send upd(adr,  $M, N, d$ ) over  $K$ 
    fi

```

```

[] rcv upd(0,  $M, N, d$ ) →
  cnet[ $N, d$ ] :=  $M$ 
end

```

5. Routing Using Mobile Addresses

At any instant, a mobile host i can use its two local variables CN and cr to send a $\text{data}(L,$

$K, c, \text{text})$ message to any other host (K, c) located in some network L . When a mobile host i receives a $\text{data}(i, M, N, d, \text{text})$ message, it knows that it is the ultimate destination of the message and stores the message text. Thus, the following two actions need to be added to the code of mobile host i .

```

action
  true →
    /* send a data msg to host ( $K, c$ ) at  $L$  */
    Define  $L, K$  and  $c$ ;
    if  $L = \text{CN}$ 
    then
      adr := arp.( $L, K, c$ );
      send data(adr,  $L, K, c, \text{text}$ ) over  $L$ 
    else
      adr := arp.( $\text{CN}, \text{CN}, \text{cr}$ );
      send data(adr,  $L, K, c, \text{text}$ )
      over  $\text{CN}$ 
    fi

```

```

[] rcv data( $i, M, N, d, \text{text}$ ) →
  store text
end

```

If the current network L of host (K, c) is unknown, the data message can be addressed to (K, K, c), because routers in network K will reroute the message to network L .

To simplify the exposition, we have not included the sender's address in data messages, although in practice it will be included. Hence, host i will learn of the address (L, K, c) upon receiving the next message from host (K, c). This allows host i to send all subsequent messages to (K, c) with the address (L, K, c).

One more action is needed in the router to forward data messages. Due to the complexity of the action, we present its code in stages. The action has the following form, where fragments AA , BB , and CC are explained below.

```

action
  rcv data ( $j, M, N, d, \text{text}$ ) →
    if  $\neg(M \in \text{SN})$  then
      AA
    else /*  $M \in \text{SN}$  */
      if  $M \neq N$  then

```

```

        BB
    else /* M = N */
        CC
    fi
fi
end

```

After receiving a data message, the router checks if the destination network M is adjacent to it. If it isn't, then it forwards the message to the next router to the destination. Thus, the code fragment AA is the following.

```

(K, c) := route.M;
adr := arp.(K, K, c);
send data(adr, M, N, d, text) over K

```

If the destination network M differs from the network in the host's name (i.e., $M \neq N$), the host is presumed to be located in network M . If M is adjacent to the router, the router forwards the message to host (N, d) in M . Thus, code fragment BB is the following.

```

adr := arp.(M, N, d);
if adr  $\neq$  0
then
    send data(adr, M, N, d, text) over M
else
    skip /*discard data msg*/
fi

```

If the destination network is the same as the network in the host's name, and this network is adjacent to the router, then the router should be aware of the host's current network. Thus, the router must forward the message to the current network of the host, which may or may not be adjacent to the router. Hence, code fragment CC is as follows.

```

if cnet[N, d]  $\in$  SN
then
    adr := arp.(cnet[N, d], N, d);
    if adr  $\neq$  0
    then
        send data(adr, cnet[N, d],
            N, d, text) over cnet[N, d]
    else
        skip /*discard data msg*/
    fi
fi

```

```

else /*  $\neg$ (cnet[N, d]  $\in$  SN) */
    (K, c) := route.cnet[N, d];
    adr := arp.(K, K, c);
    send data(adr, cnet[N, d], N, d, text)
        over K
fi

```

6. Mobile Routing Protocol

The complete mobile routing protocol can be obtained by combining the constants, variables, and actions presented in the last three sections.

In this case, the code for a host contains the two constants PN and pd , which form the name of the host. It also contains the two variables CN and cr , which form the name of a router in the current network of the host. The host has three actions: one action to generate its new address, a second action to send data messages to other hosts, and the third action to receive and store data messages.

The code for a router contains the two constants SN and SM , which are the set of networks attached to the router and the set of names of the router, respectively. It also contains the variable $cnet$, which stores the current network of each host whose name has a network in SN . The router has four actions. In the first action, the router advertises its name in every attached network. In the second and third actions, the router receives and handles address propagation messages from mobile hosts in other networks. In the last action, the router routes received data messages according to their addresses.

7. Mobile Virtual Circuits

We have thus far considered the effects of mobility on sending datagram traffic. In this section, we briefly discuss the impact of mobility on an alternative type of traffic: virtual circuit traffic.

A virtual circuit consists of two hosts, called source and destination, and a fixed path from the source to the destination. The source sends data messages to the destination, which must be routed through the fixed path. To ac-

compish this fixed routing, each router maintains information about each virtual circuit whose path includes the router.

Consider the case where a host (K, c) tries to establish a virtual circuit to a host (N, d) in a system where all hosts are stationary. First, host (K, c) sends a connect request message, $crqst(N, d, K, c)$, destined to host (N, d). When a router receives this message, it reserves resources for the circuit, such as bandwidth and buffer space. Then, the router forwards the $crqst$ message to a neighboring router according to its routing table, and stores in its memory that this neighbor is the next hop for circuit (K, c, N, d).

If a router is not able to allocate the required resources, it returns a disconnect reply message, $drply(K, c, N, d)$, along the path back to the source, causing all routers on the path to deallocate the resources reserved for the circuit. Otherwise, the $crqst$ message arrives at the destination, which returns a connect reply message, $crply(K, c, N, d)$, along the path.

The mobility of hosts requires some changes in the above scenario. Assume that host (N, d) is currently located in network M, $M \neq N$, and that host (K, c) is in network L. In this case, host (K, c) sends a $crqst(N, N, d, L, K, c)$ message to setup a circuit with (N, d). Eventually, a router in network N receives the $crqst$ message and forwards a $crqst(M, N, d, L, K, c)$ message to network M, thus building a circuit from network L to network N then to network M.

The above scheme allocates resources unnecessarily by not building the circuit directly from network L to M. This is because it only uses the $crqst$, $crply$ and $drply$ messages also used in the stationary host case. To build a direct circuit from network L to network M, other messages need to be incorporated as follows.

Any router in network N that receives a $crqst(N, N, d, L, K, c)$ message returns a forward message, $fwd(L, K, c, M, N, d)$, back to the source, indicating that host (N, d) is located in network M. The fwd message causes the

routers in the path from N to L to deallocate the resources reserved for this circuit.

Message fwd differs from message $drply$ because the source is informed that the circuit setup failed not due to lack of resources, but due to the mobility of the destination. Thus, the source may retry by sending a $crqst(M, N, d, L, K, c)$ message to build a circuit over a direct path from network L to M.

Although the above approach chooses the optimal path, it suffers from the drawback of temporarily allocating resources in the path from network L to network N until a router at N generates a fwd message. An alternative approach is for host (K, c) to send a $query(N, N, d, L, K, c)$ message to host (N, d), which in turn replies to the source with a $resp(L, K, c, M, N, d)$ message, indicating that its current network is M. The source would then proceed to build the circuit by sending a $crqst(M, N, d, L, K, c)$ message to network M.

This last approach has the disadvantage of increasing the circuit setup latency by one round-trip time in the case when host (N, d) is still residing in network N. Which of the above two approaches is superior depends on the likelihood of this case.

In the event that one host moves from one network to another while the circuit is active, the moving host may tear-down the old circuit and establish a new circuit to its peer. In the event that both hosts move at the same time and loose track of each other's current location, the source can establish a new circuit using one of the two strategies described above.

8. Mobile Groups

A group is a collection of hosts such that each message sent by a host in the group is routed to every other host in the group. A group is mobile if some of its hosts are mobile.

Associated with each mobile group is a collection of routers, called the group routers, that are arranged in an outgoing, directed, rooted tree.

The router at the tree root is called the group root, and every router is on a directed path from the group root to some host in the group. Each group router maintains a list of its parent and children in the group tree. (The parent of a group router, other than the root, is another group router, and each child of a group router is either a group router or a host in the group.)

A group, whose root is router (N, N, d) , is assigned the (mobile) address (N, N, d, g) , where g is a unique identifier of the group in router (N, N, d) .

Initially, a group (N, N, d, g) consists of only one site, namely, the root router.

A computer (L, K, c) , whether a host or a router, can join group (N, N, d, g) by sending a $\text{join}(N, N, d, g, L, K, c)$ message whose ultimate destination is router (N, N, d) , the group root. Computer (L, K, c) sends this join message to router (L', L', c') indicated by its routing table to be the best neighbor to reach the ultimate destination (N, N, d) . Later, computer (L, K, c) receives an $\text{acpt}(L, K, c)$ message from router (L', L', c') indicating that computer (L, K, c) has already joined the group and that its parent in the group tree is router (L', L', c') .

When router (L', L', c') receives the $\text{join}(N, N, d, g, L, K, c)$ message from computer (L, K, c) , it checks whether or not it is on the tree of group (N, N, d, g) . If router (L', L', c') is already on the group tree, it adds computer (L, K, c) to the set of its children in the tree then sends an $\text{acpt}(L, K, c)$ message to computer (L, K, c) . If router (L', L', c') is not on the group tree, it sends a $\text{join}(N, N, d, g, L', L', c')$ message to router (L'', L'', c'') indicated by its routing table to be the best neighbor to reach the ultimate destination (N, N, d) . Later, when router (L', L', c') receives an $\text{acpt}(L', L', c')$ message from router (L'', L'', c'') , it sends an $\text{acpt}(L, K, c)$ message to computer (L, K, c) and makes (L, K, c) its child in the group tree.

After a computer joins a group tree, it periodically sends tick messages to its parent on the tree. If a mobile host on a group tree changes its current network, it stops sending tick mes-

sages to its parent (router) on the tree. In this case, the parent recognizes that the mobile host is no longer on the group tree. Later, when the mobile host settles in another network, it can re-join the group tree using the same protocol discussed above.

Note that if a router does not receive tick messages from any of its children on the group tree for some time, the router recognizes that it no longer needs to be on the group tree, and stops sending tick messages to its parent on the tree.

To send a data message to every host in group (N, N, d, g) , a host, whether mobile or stationary, sends a $\text{data}(N, N, d, g, \text{text})$ message whose ultimate destination is router (N, N, d) , the group root. When router (N, N, d) receives the data message, it forwards a copy of the message to each of its children. In turn, each child forwards a copy of the message to each of its children, and so on until each host in the group receives a copy of the message.

9. Relationship to Current IP Addressing

Our addressing structure (M, N, d) is not supported in the addressing schemes of either the current IP, referred to as IPv4 [8], or the proposed next generation of IP, known as IPv6 [9]. Instead, addresses in both versions of IP are equivalent to our (N, d) naming structure.

In IPv4, sending to an (M, N, d) address can be emulated as: sending to (N, d) *in care of* any router in network M . This emulation requires a thin header around the sent message, which would require more bits than those needed for the (M, N, d) address.

To support our (M, N, d) address structures, IPv6 needs to provide two features. First, IPv6 needs to allow cluster addresses that can be used to route messages to any router in a given network. Second, IPv6 needs to allow multicast addresses that can be used to route message copies to every router in a given network.

Our protocols for generating, propagating, and routing using universal mobile addresses can be compared with the protocols proposed

in [5] for supporting mobility in IPv4 as follows.

1. The advertisement of a router's name in Section 3 is comparable to the advertisement by the foreign agent in [5].
2. Our protocol does not require any registration with a naming (i.e. foreign) agent. Note that we recognize that commercial and administrative issues may require this registration, but actual mobility of a host can be supported without registration with a foreign agent.
3. The propagation of mobile addresses in Section 4 differs from the mobile host registration with the home agent in [5] as follows. In our protocol, the mobile host registers its location with every router (i.e., home agents) on network N, while in [5] the mobile host registers with a specific home agent.
4. Our routing protocol is more robust in the manner that data messages are routed to mobile hosts in that any foreign agent on the host's current network is allowed to deliver data messages to the mobile host, as opposed to a single foreign agent doing so.
5. The protocols for supporting authentication, encryption, and route optimization in [5] can be added to our set of protocols defined in the previous sections.

10. Concluding Remarks

In this paper, we proposed a new addressing scheme to support mobility in a multi-network system. An address consists of a triple (M, N, d), where (N, d) is the unique name of the computer in the system, and M is the network where the computer is currently located. As a computer (N, d) moves from one network to another, its address is always unique since its name is also unique.

This addressing scheme is uniform for both mobile and stationary computers. This is because a stationary computer (N, d) can be viewed as a mobile computer that is always in

network N, and thus always has the address (N, N, d).

Because any router in the current network of a mobile host is capable of delivering data messages to the host, our addressing scheme is robust. That is, if one router in the network fails, other routers may still deliver messages to the host. Furthermore, this robustness allows for load balancing as follows. If the routing algorithm balances the traffic into the network among all routers in the network, each router may deliver the data messages directly to the mobile host.

References

- [1] D. Cohen, J. B. Postel, and R. Rom, IP addressing and routing in a local wireless network, Proc. INFOCOM 1992, pp. 626-632.
- [2] D. Comer, Internetworking with TCP/IP, Volume I, Second Edition, Prentice-Hall Inc., 1991.
- [3] J. Ioannidis, D. Duchamp, G. Q. Maguire Jr., IP based protocols for mobile internetworking, Proc. ACM SIGCOMM 1991, pp. 235-245.
- [4] A. Myles and D. Skellern, Comparing four IP based mobile host protocols, Computer Networks and ISDN Systems, Vol. 26, 1993, pp. 349-355.
- [5] IETF Mobile-IP Working Group, IP Mobility Support Internet Draft - work in progress, July 1994.
- [6] W. R. Stevens, TCP/IP Illustrated, Volume 1, Addison-Wesley, 1994.
- [7] F. Teraoka, Y. Yokote, and M. Tokoro, A network architecture for providing host migration transparency, Proc. ACM SIGCOMM 1991, pp. 209-220.
- [8] J. Postel, Internet Protocol, RFC-791, September 1981.
- [9] R. Hinden, Editor, "Internet Protocol, Version 6 (IPv6) Specification", Internet Draft, In preparation.