

8.1-5

Automatic Selection of a Visited Network in Mobile IPv6

Mariana Simons-Nikolova and Xuemei Bodlaender-Pu
Philips Research, Eindhoven, the Netherlands

Abstract— This paper describes an algorithm for automatic selection of a visited network in Mobile IPv6 (MIPv6) such that the network selected is optimal with respect to MIPv6. As a result a seamless handover to a visited network is ensured and therefore continuously running applications like MP3 playback, A/V conferencing, IP telephony and multi-user gaming.

I. INTRODUCTION

When a Mobile Node (MN) moves away from its home network and connects to a wireless visited network, one may have to choose from a plurality of networks available. Some known selection criteria for this are access point signal-strength, a random choice, or manual selection by a user. In particular, no criteria related to MIPv6 [1] are used which may result in a communication being interrupted due to lack of support for IPv6 [2] in the visited network.

The paper proposes an algorithm for automatic selection of a visited network in MIPv6 based on criteria defined by a MN manufacturer and/or user, and tuned to MIPv6 in particular. Possible criteria are: IPv6 enabled visited network, free access, no encryption, and link quality level.

Each MN, which has IPv6 and Mobile IPv6 functionality build in by its vendor when manufactured, can carry out this algorithm. It introduces two major improvements with respect to the current selection of a visited network in MIPv6: the MN stays IPv6 connected even while moving in IPv4-only visited networks; and the MN detects and connects to a visited network with optimal performance with respect to MIPv6. As a result a seamless handover to a visited network is ensured and therefore continuously running applications.

The MN home network is illustrated in Fig. 1 and consists of an IPv6/IPv4 or IPv4-only Home Router, a Home Agent (HA) that is IPv6/IPv4 and MIPv6 enabled, and a MN that is IPv6, e.g., IPv6-only or IPv6/IPv4 and MIPv6 enabled.

Various types of MN visited networks are illustrated in Fig. 1. All of them are public wireless IP networks.

II. ALGORITHM FOR AUTOMATIC SELECTION OF A VISITED NETWORK

A software component called *NetAdapter* is designed to automatically detect and connect a MN to a visited network with optimal performance with respect to MIPv6. The algorithm implemented consists of the following steps.

First, the NetAdapter detects a list of available visited networks using one of the available technologies nowadays. Second, the NetAdapter sorts this list based on predefined criteria. Third, the NetAdapter selects the first visited network in the sorted list and let the MN automatically connect to it. We allow this automatic connection to be overruled if the MN user's preference is set to manual configuration.

While situated in a visited network, the MN is associated with a *care-of address (CoA)*, which provides information about the MN's current location. The smoothness of the handover between the home network and a visited network or between two visited networks strongly depends on the MN ability to quickly auto-configure its CoA in the visited networks.

Based on our experimental results, we detect that the handover time can vary between the different visited networks. If a MN has a single network interface, e.g. eth0, then eth0 is involved in two activities during the MIPv6 handover. The first one is to support continuously running applications. The second one is to select a visited network and to auto-configure its IPv6 CoA. We propose to distribute these two activities between more than one network interfaces (if available) in order to substantially reduce the handover time and to get smooth handovers.

III. IMPLEMENTATION

The notations specified in Table I are used to describe the implementation of the algorithm the NetAdapter carries out.

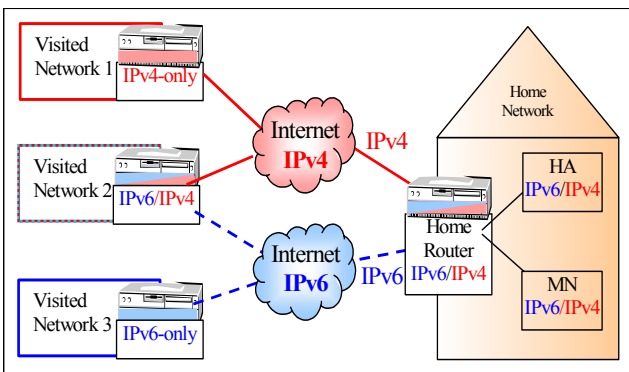


Fig. 1. General set-up in MIPv6

TABLE I
NOTATIONS USED

Notation	Description
$List_Net$	An initial list of possible visited networks generated by the NetAdapter based on one of the available technologies nowadays.
$ip_version$	An integer variable with two values: 6 or 4. If $ip_version$ is 6, then the network is IPv6 enabled, i.e. IPv6/IPv4 or IPv6-only, otherwise IPv4-only network.
$rank$	a positive integer variable
$List_Net_i(ip_version, rank)$	i -th element of $List_Net$ with two parameters – $ip_version$ and $rank$
$List_Net_Priority$	$List_Net$ sorted according to predefined criteria.
R_1, R_2, \dots, R_n	A set of rules used to generate a $rank$.
C_1, C_2, \dots, C_n	A set of checkpoints for indicating availability of certain network feature. Each C has two values: 1 or 0. If a network feature i is available, then $C_i = 1$, otherwise $C_i = 0$. The list of checkpoints is defined either by the manufacturer or by the user.
W_1, W_2, \dots, W_n	A set of positive integer values indicating the weights of the corresponding checkpoints. $W_i, i=1, \dots, n$ are defined either by the manufacturer or by the user.

Each MN contains a NetAdapter activated while MN being away from home. NetAdapter generates an initial list of possible visited networks $List_Net$. Each visited network in the list contains two parameters: $ip_version$ and $rank$.

The rank of each visited network is defined as follows.

A. Rank Definition

The $rank_i$ of i -th element in $List_Net$ is: $rank_i = \sum R_{i,j} = \sum W_{i,j} * C_{i,j}$, $j=1, \dots, n$. The checkpoints ($C_{i,j}$) and the weights ($W_{i,j}$) are defined either by the MN vendor or by the user and are stored in the NetAdapter. An example of checkpoints and weights of the i -th element in $List_Net$ is given in Table II.

The NetAdapter multiplies the corresponding checkpoints values with the weights and derives the corresponding rules $R_{i,j}$ as shown in Table II. Finally, the $rank$ and the $ip_version$ for each visited network are derived. For our example in Table II the $rank$ is 10 and the $ip_version$ is 6, and therefore $List_Net_i(ip_version, rank)$ equals to $List_Net_i(6, 10)$.

Then, the NetAdapter uses the $ip_version$ and the $rank$ values of each element to sort $List_Net$ and to generate $List_Net_Priority$. The following sorting criteria are applied.

B. Sorting Criteria

Sorting Criterion 1. IPv6 enabled visited networks have higher priority than IPv4 enabled visited networks. For example, $List_Net_i(6, rank)$ will be listed in front of $List_Net_j(4, rank)$ in $List_Net_Priority$;

Sorting Criterion 2. If the MN is IPv6-only and one of the available visited network is IPv4-only, then this visited network will not be entity of $List_Net_Priority$. For example, $List_Net_i(4, rank) \notin List_Net_Priority$ if MN is IPv6-only.

Sorting Criterion 3. Within the subsets of $List_Net(6, rank)$ and $List_Net(4, rank)$ the visited networks will be sorted with respect to the rank parameter. For example, $List_Net_i(6, 7)$ will be listed in front of $List_Net_j(6, 3)$ in $List_Net_Priority$.

TABLE II
EXAMPLE OF CHECKPOINTS, WEIGHTS AND RULES

$C_{i,j}$	Description	Value C	$W_{i,j}$	$R_{i,j}$
$C_{i,1}$	IPv6 enabled	1	3	3
$C_{i,2}$	free access	1	2	2
$C_{i,3}$	no encryption	1	2	2
$C_{i,4}$	link_quality_higher_90	0	1	0
$C_{i,5}$	link_quality_higher_70	1	3	3
...
$rank_i$			$\sum R_{i,j}$	10
IP version				6

NetAdapter sorts all networks from $List_Net$ into $List_Net_Priority$ according to the sorting criteria above. After applying these sorting criteria, the MN automatically connects to the first network in $List_Net_Priority$ unless different user preferences are specified.

IV. CONCLUSIONS

A mobile node, which implements the algorithm described in this paper, can detect and connect to a visited network with optimal performance with respect to MIPv6. Further, we propose a distribution of the MN handover activities between more than one network interfaces. Then, the primary network interface is entirely used by continuously running applications whereas a secondary network interface (if available) carries out in the background all work needed for IPv6 CoA auto-configuration of the primary network interface. The work of multiple network interfaces in parallel reduces the MIPv6 MN handover time and ensures continuously running applications.

REFERENCES

- [1] RFC 3775, Mobility Support in IPv6, June 2004, www.ietf.org/rfc/rfc3775.txt.
- [2] RFC 3513, IP Version 6 Addressing Architecture, April 2003 (Obsoletes: 2373), www.ietf.org/rfc/rfc3513.txt.