Master's Thesis

Analysis of IPv6 Neighbor Discovery for Mobile and Wireless Networks

Hariharasudan Vigneswaran
Jeena Rachel John

Master's Thesis

Analysis of IPv6 Neighbor Discovery for Mobile and Wireless Networks

By

Hariharasudan Vigneswaran
wir13hvi@student.lu.se

Jeena Rachel John
wir13jjo@student.lu.se

October 21, 2015

Advisors: Samita Chakrabarti (Ericsson AB, San José)
           Jaume Rius I Riu (Ericsson Research, Stockholm)
           Jens A Andersson (EIT LTH)
           Stefan Höst (EIT LTH)

Department of Electrical and Information Technology
Faculty of Engineering, LTH, Lund University
SE-221 00 Lund, Sweden

Abstract

The majority of the current 3GPP and M2M networks use or will use IPv6 for accessing the internet. These IPv6 networks use Neighbor Discovery as defined in RFC 4861 to identify their neighbors on the link and see if they are active. The increase in the complexity of wireless networks and introduction of battery operated devices sets forth notable challenges to certain assumptions in the RFC 4861. The RFC 4861 is more suited for wired networks and its implementation in wireless networks makes it more inefficient.

This thesis work focuses on the energy efficient implementation of RFC 4861 using the protocols mentioned in *draft-chakrabarti-nordmark-6man-efficient-nd-06*. The draft suggests a method of registering all the nodes in the network to their default router, so that the router takes care of all the basic neighbor discovery functionalities without disturbing the battery operated devices which are in sleepy mode. Alongside the legacy Neighbor Discovery, the optimizations proposed by the draft are implemented, on a RADVD based router and an Ubuntu host, to reduce redundant multicast signaling in mobile networks. These optimizations are theoretically analyzed to check if the draft is beneficial for the scalability and transient nature of the wireless networks.

Popular Science Report

With the rise of Internet of Things (IoT) and Machine to Machine communication (M2M) based networks, more and more wireless devices are communicating to each other and hence there is a need for IPv6 (Internet Protocol version 6) addressing. This new type of addressing forces us to make some design changes in the existing protocols.

Wireless devices are mostly mobile, battery operated (hence a power constraint exists) and can undergo several disconnections and reconnections while moving inside the network, whereas wired devices are less prone to network instabilities and almost always connected to a power source. Because of these differences, the protocols of wired devices are not always suitable for wireless networks. Still today's wireless devices often use protocols that were designed for wired devices and tend to be inefficient. This thesis work deals with one such protocol called Internet Control Message Protocol (ICMP), which is widely used to send control messages across the network between different nodes.

Neighbor Discovery Protocol (NDP) is a part of ICMPv6 that helps a node in the network identify its Neighbors on the other side of the link. Present generation wireless devices use the protocols defined in RFC 4861 (a common reference used by major communication companies while creating products) for Neighbor discovery. But this protocol proves to be inefficient by flooding the network with unwanted multicast control messages. Hence we analyze a modified version of the existing NDP against the legacy protocol.

RFC 4861 based 'legacy NDP' works based on multicasting. Every IPv6 host is a part of several solicited node multicast groups. All nodes with the same last 24 IPv6 address bits belong to the same solicited node multicast group. NDP is also performed based on 4 major control message packets namely, Router Solicitation (RS), Router Advertisement (RA), Neighbor Solicitation (NS) and Neighbor Advertisement (NA). RA is the packet from the router which contains all the information about the network and the router. It is flooded periodically through an all node multicast group to all IPv6 nodes.

When a node wants a particular router to send some information about the network immediately without waiting for periodic RAs, it sends an RS. Neighbor Advertisement is used by a node to send information about itself

to other nodes in the network. Neighbor Solicitation, as name suggests, is used by a particular node to request for information about another node.

Consider the case of a network with a host and a router, and a new node enters the network. It sends out RS to the all routers multicast group, and the router responds with an RA. The new host reads the information about the network and begins address configuration. Stateless Address Auto Configuration (SLAAC) is the most suitable method of address configuration for M2M networks. Hence according to SLAAC, the new host chooses an IP address and checks if the address has already been used by some other host. This process is called as Duplicate Address Detection (DAD) and is carried out by sending out NS for that chosen IP address at the solicited node multicast group address. This means that the new host asks information about a node with that IP address, which may or may not exist at all. If a reply comes, it means that there is a node using this address. If a reply does not come, then it is free to choose the address. The response NA in case of a duplicate address, will be broadcast to all IPv6 nodes. A NS-NA pair is also used to resolve each other's IP addresses against their MAC addresses.

One of the major reasons why this method seems to be inefficient is that current wireless devices are programmed to be in sleep modes when not in use. The flow of control messages to unrelated nodes (because of multicast) will only lead to disrupted sleep. Consider a host A in the network which is in sleep mode, and has an IP address aaaa::aaaa:aaaa:a. When a new host arrives and wants to register itself with an address of aaaa::baaa:aaaa:a, it sends out NS to the multicast group of A (because A has the same last 24 bits). When a packet reaches A, it wakes up from sleep mode, reads the message and identifies that this message was not destined to it. If there was another host in the group which used the same IP address, then host A will also receive another NA, which again is not of any use. Hence this protocol proves to be inefficient.

The new proposed protocol works with an efficient implementation of the NDP. It introduces a concept of registering a host with a router. And the router sends out NS and NA instead of the host. Hence each router's NCE will be the source of information for those hosts which are registered to that router. If a node will request for details about another host, the details will either be fetched from that router's NCE or the NS will be forwarded to that particular host instead of being multicast. The router to whom the

6

destination host is registered will then send NAs on behalf of its host. Hence in this protocol, the router performs more work and the host less, allowing the host to stay in sleep modes without any problems.

The analysis of the two protocols begins with their implementation. We decided to test the working of the two protocols at two specific case scenarios: when several hosts join the network and when the channel links are unsteady. Both protocols were implemented and compared. The implementation of the legacy NDP began with a router and a host system. We used an Ubuntu PC (which generally works as a host) as a router. RADVD is an open source software that helps to generate RAs and Wireshark the software used to analyze packets in the network. To make a host work as a router, the following changes were made:

1. Turn on forwarding in the PC
2. Configure the router with a static IP address
3. Configure the interfaces and add corresponding routing information
4. Install RADVD and include all the network parameters in the config file
5. Turn on Wireshark
6. Turn on RADVD

Once all these steps are done in order, the RADVD sends out periodic RAs that can be viewed on the Wireshark. When the host is connected to the network, the entire process of address configuration and DAD can be identified. When pinged, the address resolution process can also be verified. To implement the new protocol, we downloaded the source code of the open source RADVD from GitHub and change the packet formats of the RA, NS and NA.

When the router works as a NEAR (Neighbor Discovery Efficiency Aware Router), the host has to understand that it should not use legacy NDP anymore and must act as an Efficiency Aware host (EAH). This information comes with the RA in the form of a flag. When this is done, the host sends an NS in the form of registration request. The details of the host go in the Address Resolution Option (ARO) of the NS. The router responds with an NA which is the acknowledgement for the registration request. Now the host can enter into sleep mode and the router will send control packets to it only when required. When the registration expires, a new registration process has to be setup or a de-registration process has to happen by setting the ARO option in the NS to 0.

To implement the changes in the host side, we downloaded the source code for Linux from GitHub and led the decoder side. The process ends by making all the necessary code changes, recompiling a new Kernel and installing the new version of Linux on the hosts. However, the complete reprogramming of the kernel is not handled in this thesis work. But the entire plan for the programming and the rest of the implementation is included in this report. The theoretical analysis of the protocols is also included in this report.

This report concludes with a general analysis of the whole protocol. The whole protocol which was completely decentralized has a certain degree of centralized control now. As with all systems, centralization introduces better control but there is always a problem of device failure which can completely collapse the system. The protocol also provides solutions for router failure and recovery and hence the system is a good compromise between centralized and decentralized systems.

If properly implemented, there will be an increase in the network efficiency (inversely proportional to the number of control packets for every single data point). This thesis ends with the theoretical analysis. A proper implementation followed by a practical analysis would be a natural future amendment to the project

Acknowledgments

Acknowledgments

Table of Contents

Preface

This thesis work was performed by Hariharasudhan Vigneswaran and Jeena Rachel John. The initial parts of the project were worked upon together by both the member of the team. This includes literature survey of a few internet drafts and the RFC 4861, implementation of the legacy NDP and planning for the documentation. The second half of the thesis was forked into two. The implementation of the efficient ND as per the draft was performed by Hari which includes the implementation plan and procedure for the modified protocol. The theoretical analysis of the draft was performed by Jeena which gives a clear idea of the empirical results of legacy NDP for a basic wireless network and a theoretical evaluation of these results for the draft NDP.

CHAPTER 1

1 Introduction

The forthcoming rise of the Internet of Things (IoT) or Machine to Machine (M2M) networks foresees the need for interaction or communication between billions of devices. These networks use Internet Protocol version 6 (IPv6) protocol in the layer 3 as the address spacing of this protocol is tremendously wide and can accommodate billions of devices. The IPv6 protocol is based on the Neighbor Discovery (ND) mechanism as per the [1]. Neighbor Discovery is supported by the exchange of Internet Control Management Protocol version 6 (ICMPv6) signals. These ICMPv6 messages include unnecessary control signals which cause significant increase in the processing power and network traffic. The major challenge is to optimize the Neighbor Discovery Protocol (NDP) to minimize these unmerited control signals and improve the efficiency of M2M /IoT Third Generation Partnership Project (3GPP) networks. The [2] proposes certain optimizations in the base NDP to provide incremental benefits.

1.1 Background

The internet has grown and changed continuously in the past decade. Networks have become complicated with the introduction of M2M and IoT and hence more efficient methods of ND is required. The [1] is the updated protocol reference for [17] which defines how a node in the network identifies its neighbors on the other end of the link. This involves sending a few control messages which are commonly called as ICMP messages. ICMPv6 works on the principle of ipv6 multicasting. The [1] works best for wired networks and when extended to wireless network faces problems of inefficiency.

A few Internet Drafts have suggested the inefficiency of the [1]. The [3] proposed by Ericsson, highlights the EPS network impact of IoT/M2M IPv6 connectivity and [4] proposed by a team of CISCO members, highlights the issues of using multicasting in some wireless networks including Wi-Fi. The [2] suggests a registration method of using the routers

in the network as the core and converting all the multicast ICMPv6 messages to unicast. This solves the issues in both the above mentioned internet drafts. This thesis work is about implementing the efficient ND draft and analyzing if it can produce results in real time networks.

1.2 Overview

The thesis work at a glance can be described as in Fig. 1 which shows the existing model as per [1] that is non-optimized and Fig. 2 which is the optimized model as per the draft proposal.
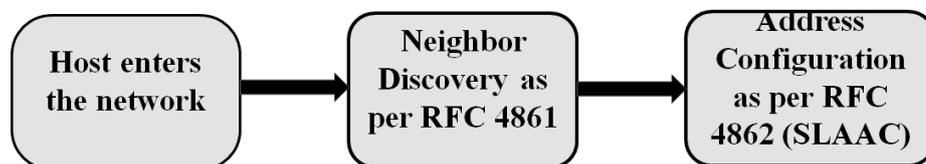
Fig. 1. A Conventional model of Neighbor Discovery Protocol as per RFC 4861[1]
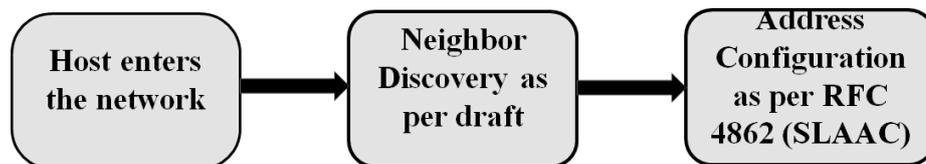
Fig. 2. Optimized model of Neighbor Discovery Protocol as per efficient-ND draft [2]

The draft is proposed to work better with current day battery operated devices (in idle mode). By reducing the multicast based Neighbor Discovery to unicast mode, the effect of the second assumption made in [1] is comparatively reduced. This energy efficient draft supports the use of SLAAC in wireless networks.

The key modifications the draft proposes to optimize [1] are:

- Removes multicast signaling
- Introduces the concept of sleepy hosts
- Efficient Duplicate Address Detection
- Increases robustness of the network

1.3 Problem definition

As already mentioned in section 1.1, the [1] has some challenges with certain types of networks which are inefficient with multicast signaling and present generation mobile devices which are battery operated. Because most of the ICMPv6 messages are multicast, there is unwanted flow of information to members which are not related in the communication, creating a problem of inefficiency. When a battery operated device (in sleep mode), is a part of a multicast group, it will receive all messages sent to the group. When the size of the group is big (as is generally the case with IoT and M2M networks) these messages are most often irrelevant to the node, but it still has to wake up from sleep, decode the packet and identify that it was not relevant. Also when the nodes enter and leave the network often which also initiates multicast messages that disturbs the devices during their sleep cycle. Thus the NDP discussed in [1] does not accord to present day hardware. Hence the modified draft has been presented for a replacement to the original RFC, which is currently the only standard that is implemented in 3GPP devices. Detailed information about how the RFC performs ND and what the problem is, will follow later in this document. Thus the goal of this thesis work is to analyze if the draft is beneficial for this scalability and transient nature of the network.

1.4 Approach

The project was planned for implementation on Linux based Ubuntu environment. RADVD is the open source that is used to send out RAs and convert a PC into a router. Hence a normal Ubuntu box was connected to two other hosts on a wireless network and the NDP was tested with different case scenarios. The whole system was also implemented on virtual box as well and some details of this environment is included in the Appendix 1. A theoretical analysis of the signaling traffic for certain network scenarios are made for testing the effect on scalability and transient nature of the network and these tests are based on certain parameters like router lifetime, router interval, etc. The complete analysis would determine the network efficiency in handling the network traffic.

1.5 Organization of the Thesis

This thesis work is organized as follows: Chapter 1 introduces the thesis and the actual problem we face in the current protocols. Chapter 2 provides an overview of the ND protocol as per [1] and how the address resolution, Duplicate Address Detection is defined and the limitations of the RFC are

highlighted to point out the beneficial optimizations of the efficient ND draft. Chapter 3 introduces the new optimized ND protocol based on the *[2]*. Chapter 4 describes the implementation details of the legacy NDP and evaluates the performance. Chapter 5 provides an analysis of the implementation details of the state change of router and hosts according to the efficient-ND. Only a study of these implementations was performed on some areas in this work. The Chapter 6 makes a theoretical analysis of the legacy NDP and the efficient NDP and Chapter 7 will analyze the expected results of the assumptions made in Chapter 6. The Chapter 8 concludes the thesis work and summarizes the results and Chapter 9 gives a brief idea about the future work related to this thesis work. The Fig. 3 gives the complete structure of this thesis work.
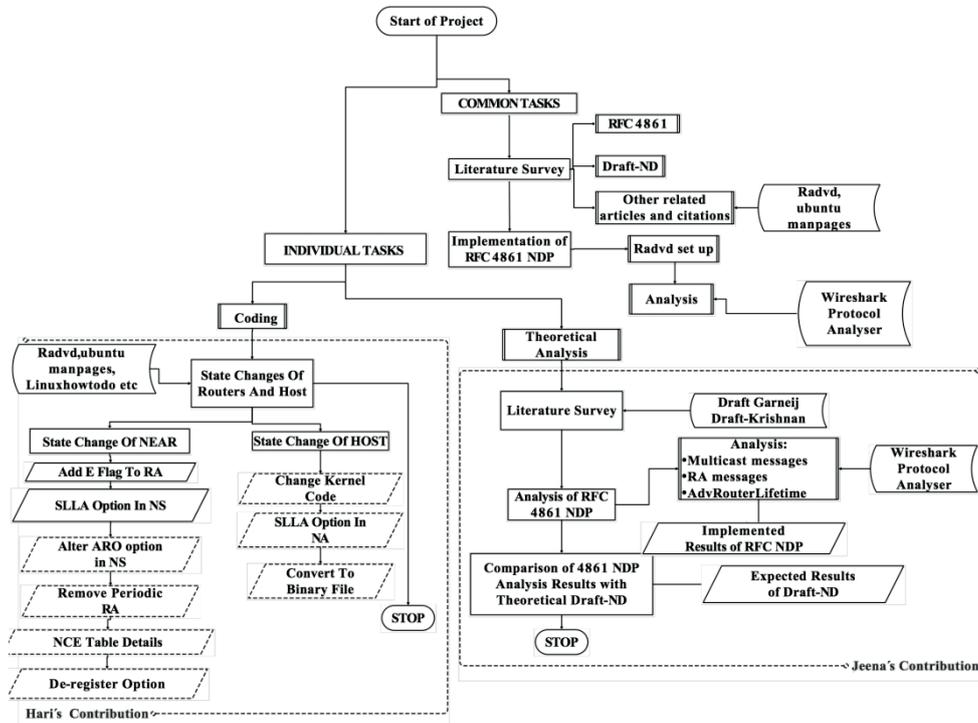
Fig. 3. The complete structure of the project plan

2 Protocol Overview of RFC 4861

Neighbor Discovery is the process by which a node (host/router) in a network gets to know about its neighbors on the other end of the link. This neighbor discovery includes each node having a list of all the neighbors, with its corresponding logical and physical address, identifying which of these neighbors is a router and identifying the network id while trying to configure itself a new IP address. The [1] provides protocols that cover the following functionalities:

1. Router Discovery
2. Network id and Address Configuration Mechanism identification
3. Address Resolution
4. Duplicate Address Detection
5. Updating of a change of MAC address to the other neighbors
6. Neighbor Unreachability

2.1 Neighbor Discovery Protocol

Neighbor Discovery is a sub-protocol of ICMPv6. Hence similar to IPv4, the NDP involves the following 5 basic packets:

1.   Router Advertisement
2.   Router Solicitation
3.   Neighbor Solicitation
4.   Neighbor Advertisement
5.   Redirect message

These packets have the formats as shown in, Fig. 4, Fig. 5, Fig. 6, Fig. 7 and Fig. 8.

Router Advertisements are packets that are broadcast out from the router every 200s. They contain information about the network including the

network id, the type of address configuration, and information about the default router.

Fig. 4. Router Advertisement Packet Format [1]

Fields [1] *in the RA Packet:*

Type:	*134*
Code:	*0*
Checksum:	*The ICMP checksum*
M:	*Managed Address Configuration Flag*
O:	*Other Configuration Flag*
Reserved:	*6 bits. This field is unused.*
Cur Hop Limit:	*8 bit. Hop count field of the IP header for outgoing packets.*
Router Lifetime:	*16 bit. Lifetime of the default router*
Reachable Time:	*32 bit. Time that a node assumes that a neighbor is reachable after the last reachability detection*
Retrans Timer:	*32 bit. Time between the retransmitted NS messages*

Router Solicitations are packets that are triggered by the nodes to the routers in request of a RA packet. A simple scenario of the use of Router Solicitation is as shown. Suppose a RA is sent out by the router at every 0s, 200s, 400s, etc., and a new host joins the network at 1s, it can trigger out a RS instead of waiting for another 199s. The RA that comes out in return is called as solicited RA.

22

Type	Code	Checksum
Reserved		
Options...		
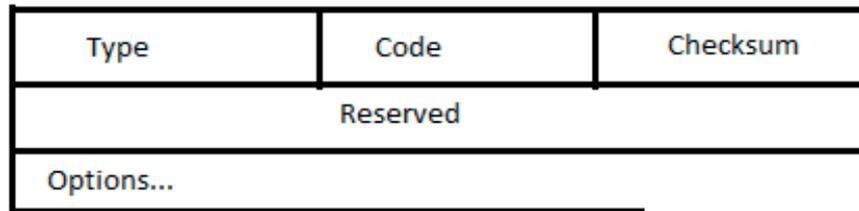
Fig. 5. Packet format of Router Solicitation [1]

Neighbor Solicitation is the packet used for obtaining some information about a node. NSs are used by a node to know the MAC address of the destination and also to identify if a neighbor is still reachable and for DAD.

Type	Code	Checksum
Reserved		
Target Address		
Options ...		
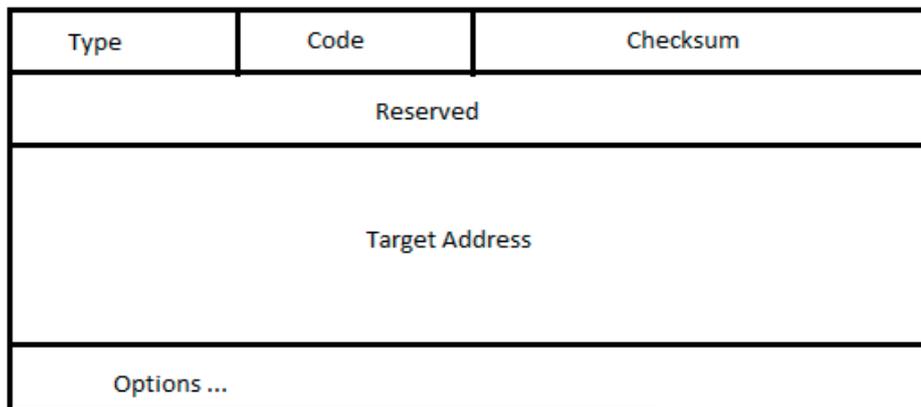
Fig. 6. Packet format of Neighbor Solicitation [1]

Neighbor Advertisement is the packet that is sent out by a node in response to the Neighbor Solicitation. Unsolicited NAs can also be sent to announce link-layer address change.
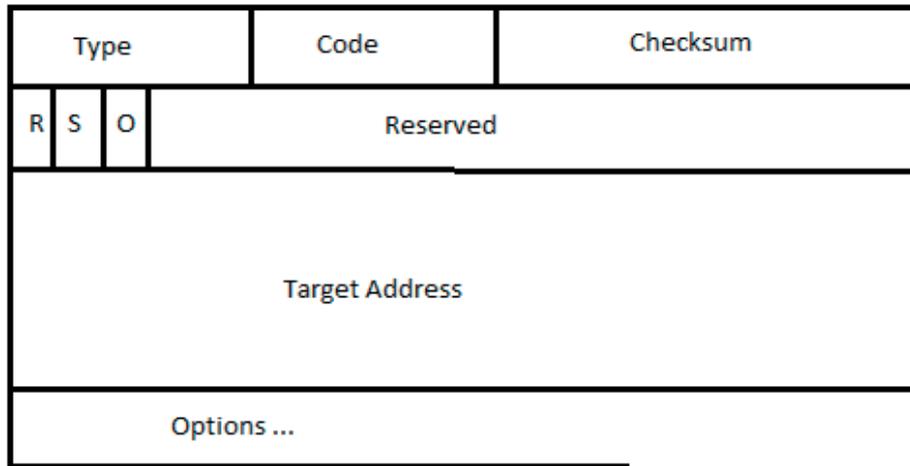
Fig. 7. Packet format of Neighbor Advertisement [1]

Redirect messages are used by routers to instruct a node about a better first hop for its destination.
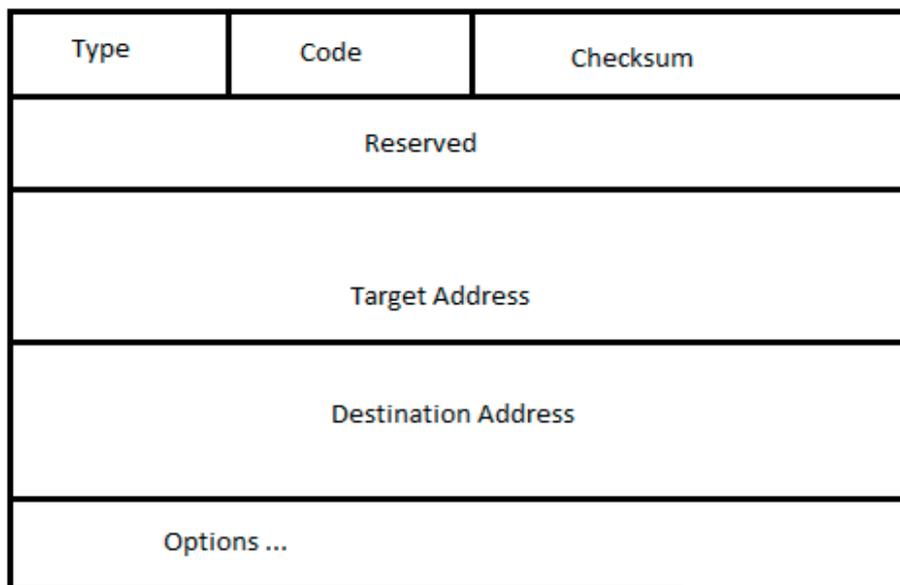
Fig. 8. Packet format of Redirect [1]

2.2 IPv6 Multicasting

The whole concept of RFC 4861 NDP is based on IPv6 Multicasting. Every IPv6 node in the network is a part of a particular 'solicited node multicast group' which is based on its IPv6 address. ICMP packets are sent to these multicast addresses all nodes. All nodes that follow IPv6 protocols are automatically a part of the FF02::1 multicast group. This is the IPv6 equivalent of the IPv4 broadcast group. All routers are a part of the FF02::2 multicast group.

Based on the IPv6 logical address, each nodes joins its solicited node multicast group. The first 104 bits of the multicast IPv6 address is FF02::1:FF (All multicast groups address begin with FF). The last 24 bits are the same as the last 24 bits of the host id.
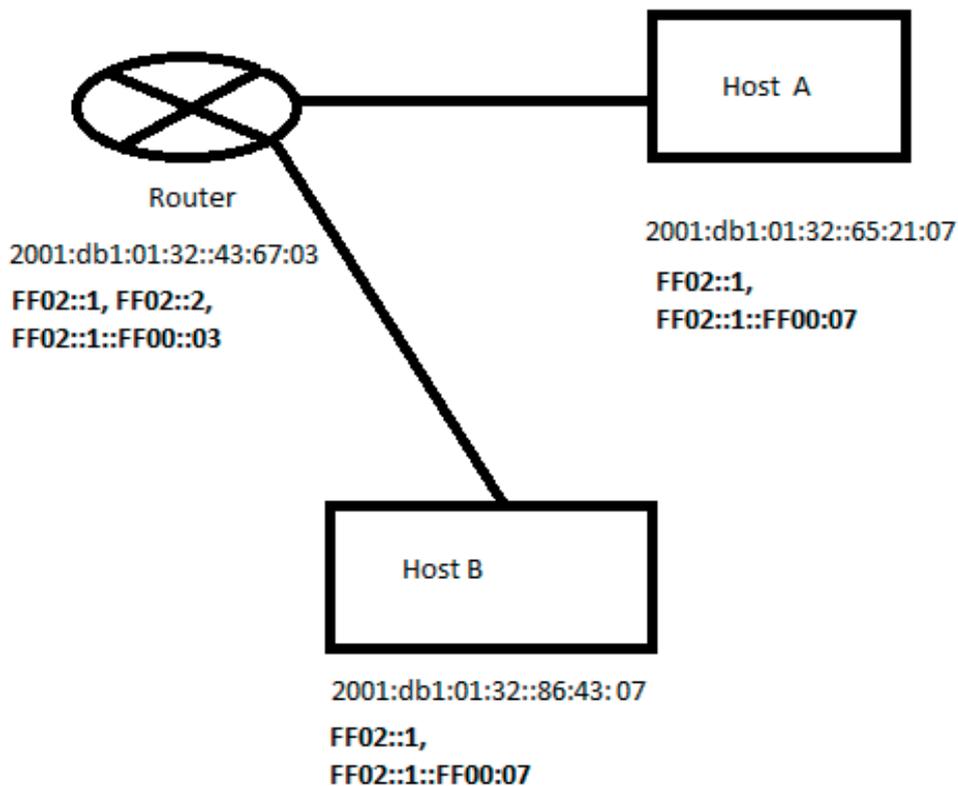

Fig. 9. IPv6 nodes with their multicast groups

The Fig. 9 is the simple scenario of a two host and one router IPv6 network. All the three nodes have the same network id 2001:db1:01:32 because they are a part of the same network but each of them have a unique host id. All the IPv6 nodes are a part of the all nodes IPv6 multicast group FF02::1. Besides the router is a part of the all routers multicast group FF02::2 as shown in Table 1.The router is a part of the FF02:1::FF00:03 solicited node multicast group. Both the hosts happen to have an identical last 24 bits of the host id and hence are a part of the same solicited node multicast group.

TABLE 1. IPV6 MULTICAST GROUPS

Multicast group	Description
FF02::1	All IPv6 nodes multicast group
FF02::2	All routers multicast group
FF02::1:FFxx:xxxx	Solicited node multicast group for a node with xx:xxxx as the last 24 bits of the IPv6 address

2.3 Router and Prefix Discovery

Router Discovery is used to locate routers in the network and identify configuration parameters related to address configuration. Prefix Discovery is the process through which hosts learn the ranges of IP addresses that reside on-link and can be reached directly without going through a router. Routers send RAs that indicate whether it is willing to be a default router. RAs also contain Prefix Information options that list the set of prefixes that identify on-link IP addresses. Stateless Address Auto-configuration must also obtain subnet prefixes as part of configuring addresses.

When a new node enters the network, it either gets unsolicited RAs or sends out a RS for a RA. The RAs contain information like *Managed Address (M) Flag* and *Other Configuration (O) Flag*. It also includes the interface's configured values like *CurHopLimit*, *Reachable Time*, *Retrans Timer*, and *Router Lifetime*. RAs also include the default MTU size and the Source Link Layer Address of the router's interface.

Fig. 10. Router and Prefix discovery

An RS is sent out to the all routers multicast group, to which all routers flood RAs to the all nodes multicast group as shown in Fig. 10. Based on all the RAs from different routers, the new host decides its default router, address configuration mechanism, network prefix and the on link neighbors. Address configuration takes place through SLAAC in general in accordance to RFC4862.

Managed Address (M) Flag is set only when addresses are managed via Dynamic Host Configuration Protocol (DHCPv6). *Other Configuration (O) Flag* is set when there is more information about the network via DHCPv6. All through this thesis SLAAC as per [2] is used as the address configuration mechanism and hence these flags are irrelevant. *CurHopLimit* defines the value that should be placed in the hop count field of outgoing packets. *Reachable Time* is the amount of time a host assumes that its neighbor (which sends this packet) is reachable. *Retrans Timer* is the retransmission time between two NSs. *Router Lifetime* is the lifetime associated with a default router.

2.4 Router and Prefix Discovery

The process of Address Resolution is another main task of the RFC 4861. A node needs to know both the MAC address and the IP address of a destination to send packets. The ICMPv6 messages help to resolve this problem. When a node wants to know the MAC address of another node, it

27

sends a NS packet to its destination solicited node multicast group. All nodes in this multicast group will receive the NS and the particular destination node will respond back to the NS with a NA. The NA will be sent to the all nodes multicast address if the NS did not have a source address. If the source address was specified in the NS, the response is sent back as unicast to the corresponding logic address. The NA packet contains the packet of the sending packet. If the NS packet was sent with proper source link layer address, both the nodes will be able to resolve each other using a single pair of NS-NA.

Host B   Router   Host A

Decides to send a packet
to Host A,but has no
entries of it in the NCE

It creates a new entry
for Host A and marks it
'incomplete'

                    NS to the solicited
                    node multicast group
                    of Host A

                                                    NA to FF02::1 from
                                                    Host A if SLLA was not
                                                    included in the NS

Updates its NCE as 'reachable'
Ideitifies the MAC
address and other
details of Host A
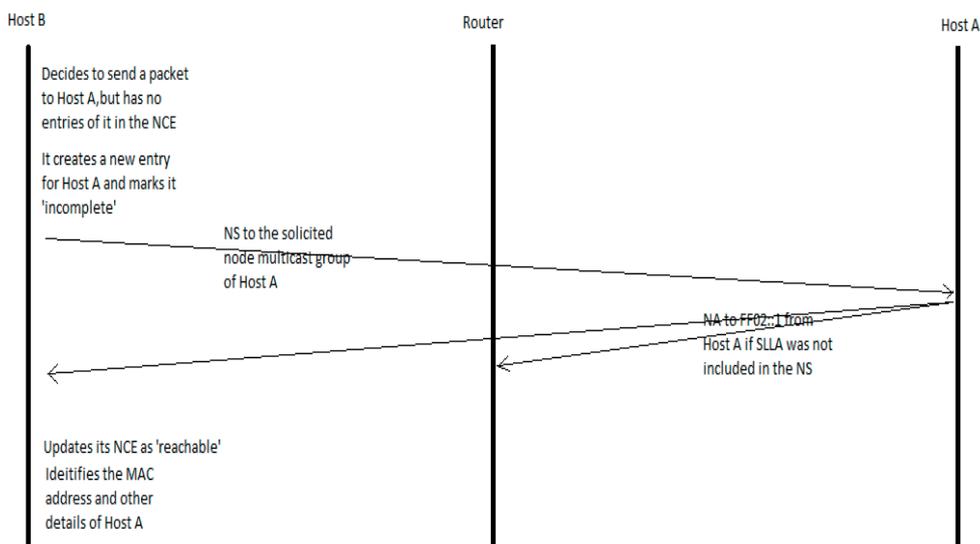
Fig. 11. Address Resolution

The actual resolution process begins with the sender checking its Neighbor Cache Entry table for information about its destination. When there is no entry for that particular destination, the sender creates a new entry with a status 'incomplete'. The process of sending NS and NAs begin. Once the sender gets the NA from the destination, it updates its NCE to 'reachable' as shown in Fig. 11. The time period for which the NCE will not require an update depends on the time set on the NA packet.

2.5 DAD

Duplicate Address Detection is the process that prevents a host from configuring itself with an IP address which is already being used by another

node. These problems are pretty common when SLAAC is used and the RFC 4861 works together with [8] in dealing with these problems.
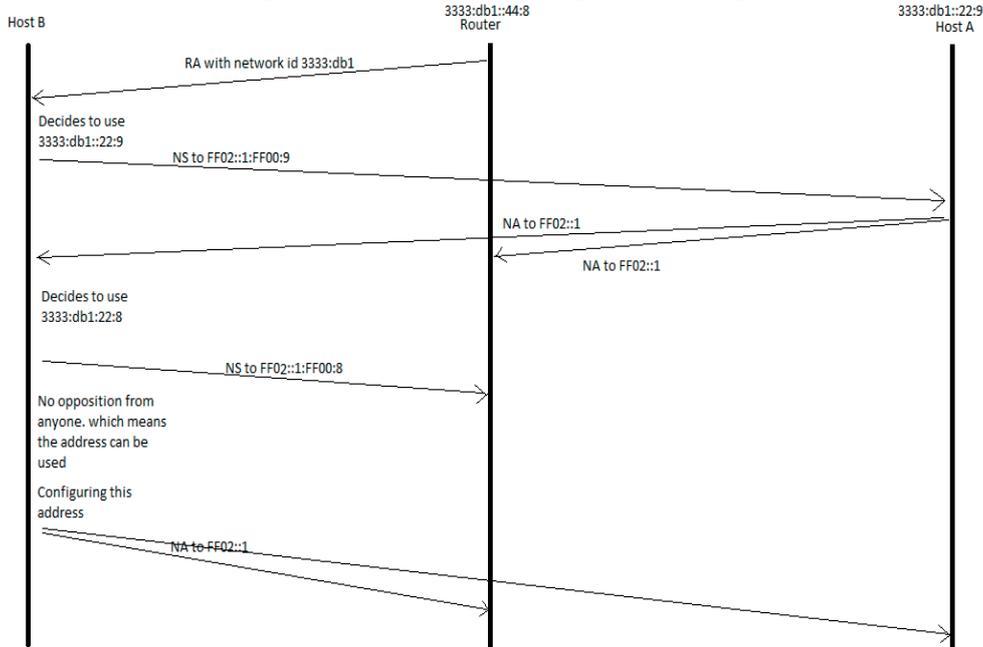

Fig. 12. Duplicate Address Detection

The Fig. 12 clearly explains what happens when a host enters a network and tries to configure itself with a new IP address. Consider a network with a host and a router, and a new host enters the network. It receives RAs from the router, and identifies the network id. Using SLAAC, the host chooses its own IP address. Let us assume that the router chooses an address already chosen by host A. It sends a NA to the solicited node multicast group of that chosen address to verify if there is someone who already uses the address. Host A which is a part of FF02::1::FF00:9 will receive the NS for its address. Since host A receives a NS from an unknown sender (Host B does not have an IP address yet. So the NS's source address is set to 00::00), it sends a NA to all nodes multicast group FF02::1. Now that host B gets the NA, it knows that someone already uses its choice, it changes its host id and tries to reconfigure itself with the new address. The same process of sending a NS to the solicited node multicast group FF02::1:FF00:8 continues. Since no one in the network uses this address, it configures itself with that address and updates all other nodes in the network about itself via a NA. This will enable other nodes in the network

29

to update their NCEs. When host B wants to send packets to other hosts, it will begin the process of address resolution to update its NCE and then transfer packets.

2.6 Limitations

As explained in the above sections, the RFC 4861 plays an important role in defining how neighbor discovery works with ipv6 networks. But these protocols were developed when the concept of wireless networks did not exist at all and these protocols work fine for fixed wired networks. But these protocols are again used in wireless networks and hence are inefficient. Besides others, two major assumptions that were made while implementing the same protocols in wireless networks were:

1. The link costs for unicast transmission and multicast transmission are assumed to be same
2. Mobile hosts always remain ON
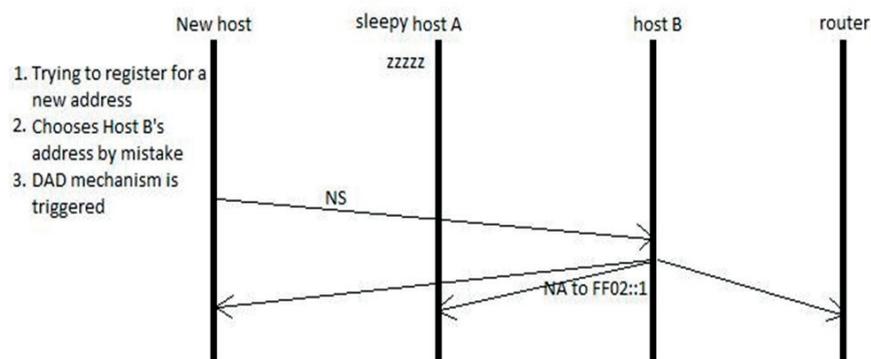
Fig. 13. Limitations in RFC 4861

Both these assumptions are no longer true for wireless networks. Battery operated devices enter into a 'sleepy' mode when they do not have packets to transmit or when they are not receiving packets actively. Sleep mode in modern day devices is when the devices try to conserve energy by adopting a duty cycling technique by either shuts down its radio (Normally OFF state) or constantly listens to the incoming packets (Always ON state) or in Low power state [14] depending on the frequency of communication associated with the device. This energy conservation mechanism is decided by the vendors. Some networks do not support multicast traffic and are

more suited for unicast traffic, in which case, it will be impossible to efficiently implement these protocols.

For example, consider a network with 2 hosts and a router as shown in Fig. 13. All the hosts are battery operated (e.g. a real world mobile phone network) and to conserve energy they enter into a sleep mode when they are not actively sending or receiving packets. Here host A is in sleep mode while a new host enters the network. When the new host tries to configure itself with B's address, DAD mechanism is triggered and host B sends NA to all nodes in the network. Even though the entire process has nothing to do with host A, it wakes up from sleep, reads the packet, and understands that it was of no use to it and then discard the packet. This is how the RFC 4861 seems inefficient for wireless networks.

3 Optimizations of Efficient draft-ND

The [2] is an efficient draft which optimizes the conventional IPv6 Neighbor Discovery defined in [1]. It extends the Neighbor Discovery mechanism to weed out the unnecessary multicast traffic. The sleepy host that exist in the current Mobile network scenario which is also discussed in the [7] makes a considerable change in the preliminary assumptions made in [1]. This draft proposes to establish those sleeping hosts to be more efficient.

3.1 Goal of the Draft

The main goal of the draft is to reduce the multicast based Neighbor Discovery traffic on a link as much as possible to unicast mode. This would comparatively reduce the effect of traffic related assumptions made in [1]. The draft is proposed to work better with current day battery operated devices (in idle mode).This energy efficient draft also provides a better strategy to handle the Duplicate Address Detection (DAD) which removes the need for the hosts to be always awake to protect their address. These optimized solutions are proposed to be independent of the address configuration mechanism.

3.2 Protocol Overview

The fundamental ND protocol optimizations proposed in the draft from the original ND as per [1] would be the registration of the nodes with its default router(s) with the Address Registration Option (ARO). These registered addresses would be used further to perform the address resolution and DAD. Thus these operations would be unicast instead of multicast messages. The periodic update messages are avoided. The ND operations would support the legacy mode, the efficiency aware mode and the mixed mode. This thesis work mainly focuses on the efficiency aware mode.

The host performs the legacy ND when it powers on and when it receives a RA from a NEAR it finds the default NEAR among the NEARs that send the RAs according to the priority set for the RA. Then it registers itself to the default NEAR to become an EAH. The EAH would refresh its address with its default NEAR before it go to sleep and need not wakeup to defend its address as the default NEAR will handle the DAD with the ARO option.

In case of conventional networks, the routers send periodic RAs to the hosts via multicast and the hosts read this information even if they are in idle mode. As per the draft, when an idle host wakes up, it sends a unicast Neighbor Solicitation (NS) to the NEAR, and the NEAR responds with unicast RA. No unsolicited RAs are sent as long as the EAH requests for an RA. Hence the battery life of the EAH increases.

When the EAH generate an IPv6 address it sends a unicast RS with the ARO option to the Address Registrar routers. The ARO option in the unicast Neighbor Solicitation is an extension to the one used in [7] with new options like *Unique Interface ID* (UID) and *Transaction ID* (TID) for the particular EAH. The NEAR checks for Duplicate address from the Neighbor Cache entry maintained which has the already registered addresses.

The hosts acquires address registration with the registrars which are normally the default routers. The draft proposes a new method of registration of addresses in the case of the loss of state for a NEAR. The *Router Epoch mechanism* is proposed to trigger the host to allow the router to rebuild the address registrations once the router initializes its state back. The Registrar Address Option (RAO) is used in this mechanism and RAO has the IPv6 registrar addresses to handle this situation. The Router Epoch mechanism is out of the scope of this thesis work.

3.3 State changes of Router and Host

The draft proposes to make some fundamental changes in the base Neighbor Discovery (ND) percept. The address resolution no longer requires multicast signaling as a set of routers would handle the address registrations of the hosts. The periodic multicast signaling to refresh the information is changed to unicast solicited signaling. To perform these operations the states of the router and hosts need to be changed to an efficiency aware mode. The router that implements the ND optimizations are known as IPv6 ND-Efficiency Aware Router (NEAR) and a host that

optimizes the ND will be an Efficiency-Aware Host (EAH). The new *E flag* field in the Router Advertisement (RA) send from the NEAR will indicate the host that the particular RA is from a NEAR as seen in Fig. 14.Thus the host registers itself with the default NEAR and changes its state to EAH.
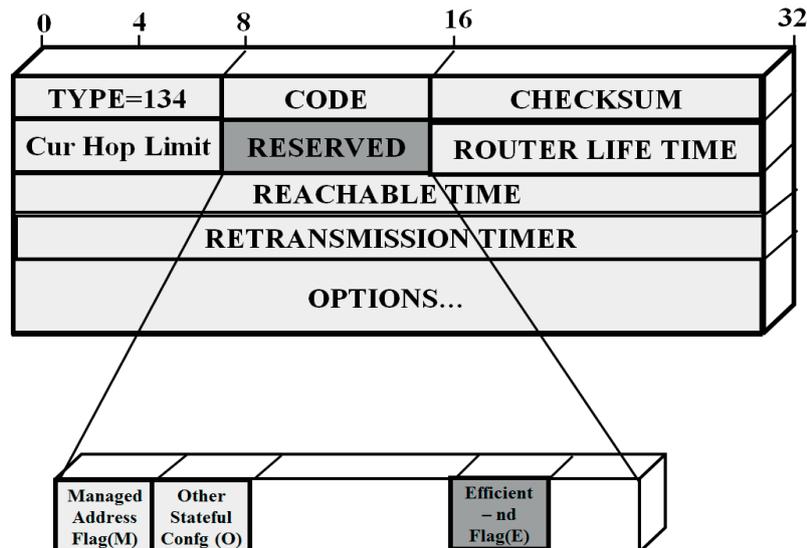

Fig. 14. Router Advertisement format for NEAR with the new E flag

3.4 Router Behavior

The NEAR would support legacy mode, efficiency aware mode and mixed mode but the thesis would implement the efficiency aware mode. During the system start up, the NEAR performs the legacy ND by sending some initial RAs and its updates as shown in Fig. 15. Besides, they are also a part of the all- nodes and all-routers multicast address group. The draft [2] proposes that the exchange of information updates from the NEARs always happen as unicast RAs in response to the RSs which is different from the case in [1] which recommends the multicast RAs for periodic updates. The unicast RAs are made possible as the RS as per draft would always compulsorily include the *Source Link-Layer Address* (SLLA) option. This option was not made compulsory in [1]. This optimization can vastly reduce the amount of unwanted signaling within the link and conserves the network resources.

When the NEAR has a new piece of urgent information to share such as new prefix added or prefix that should be removed it might multicast these

messages. But those hosts are in sleepy mode will not receive these information. So they need to solicit for these updates once they wake up before their registration expires. Thus NEAR handles the DAD and avoids the need for the sleepy hosts to wake up to protect their addresses.
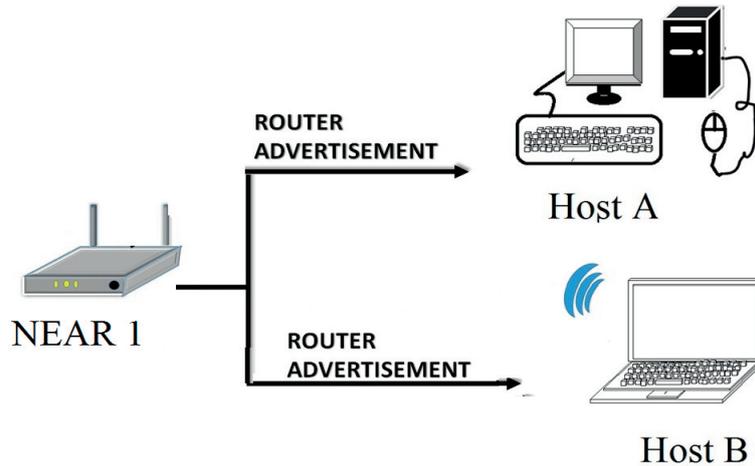
Fig. 15. Router Behavior when it's powered on or initialized

3.5 Host Behavior

A host when enters a network at its system start up or to initialize its interface, it behaves as a legacy host and multicasts Router Solicitations and its updates as shown in Fig. 16. The router on the link respond with router advertisements as unicast messages to the host which had send RS as shown in Fig. 17. The host needs to join the all-nodes multicast address group but it need not join the solicited-node multicast group.
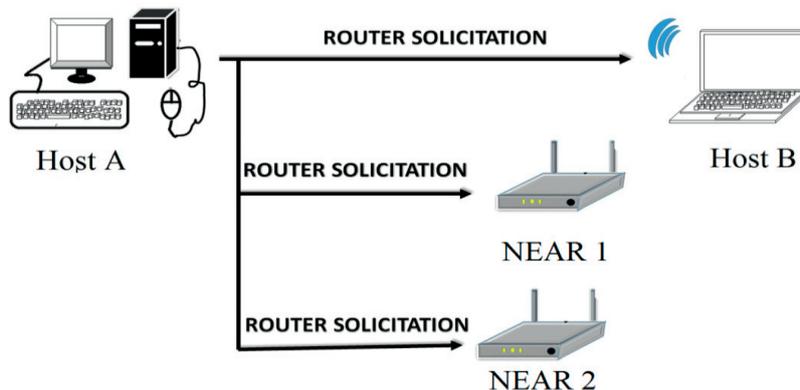
Fig. 16. Host Behavior when it's powered on or initialized
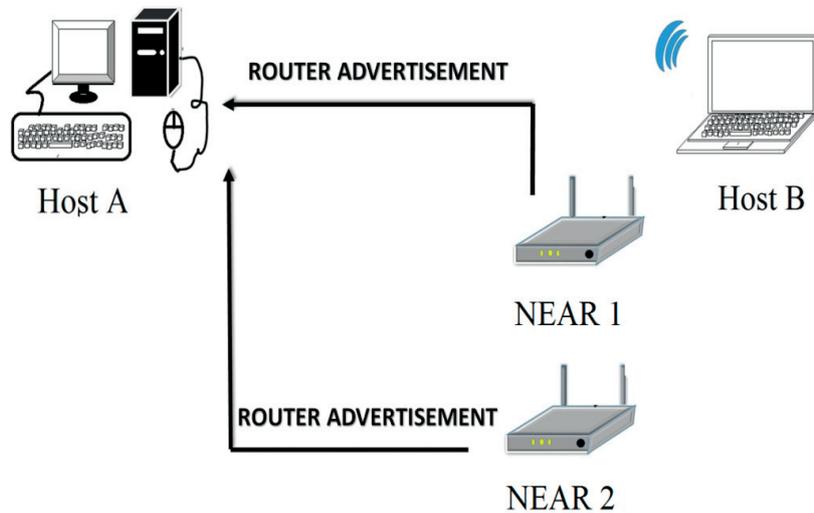
Fig. 17. The response from the NEAR routers for the router solicitation from the host

If the interface initialization is as a result of its movement or wake up from the sleep mode, the host being an EAH can send unicasts Neighbor Solicitations to the default NEAR as shown in Fig. 18.
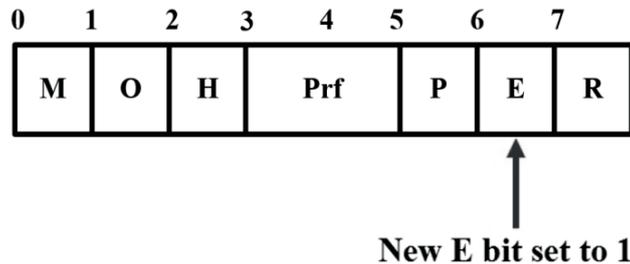
Fig. 18. The new E bit added to the RA flag field [2]

Now the host have the information about the NEARs on link which are also the address registrars. The new E flag in the RA as shown in Fig. 18 would indicate the host that the routers are NEAR and hence thereafter the host is an Efficiency Aware Host (EAH).With priority indications defined in each of these RA, the host makes a list of IPv6 address registrars and makes decision to assign a default NEAR. This list of IPv6 address registrars is called *Neighbor Cache entry* (NCE). The registrar entries have an explicit registered lifetime. These entries are kept in the NCE until the lifetime expires. This lifetime is associated with the Default Router (NEAR) lifetime indicated in the RA.

The host would perform the address allocation process now and hence instead of sending multicast DAD messages it sends the Neighbor solicitation to the default NEAR. The default NEAR gives response as NA which handles both the address registration and the DAD procedure which is shown in Fig. 19.
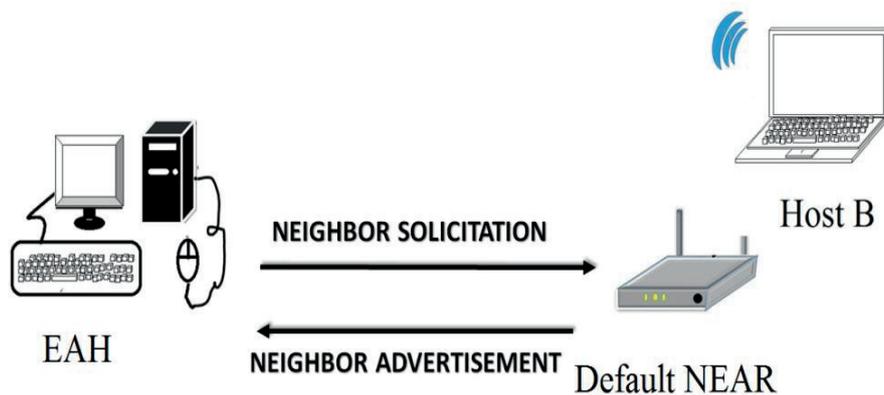
Fig. 19. Host Behavior when it makes address registration and DAD

3.6 Address Registration

The unicast Neighbor solicitation send from the EAH with SLLA option includes an Address Registration Option (ARO) which helps the NEAR which receives this NS to maintain its NCE. This ARO option was defined in [7] section 4.1 and this draft will reuse this option with extended fields as shown in Fig. 20.
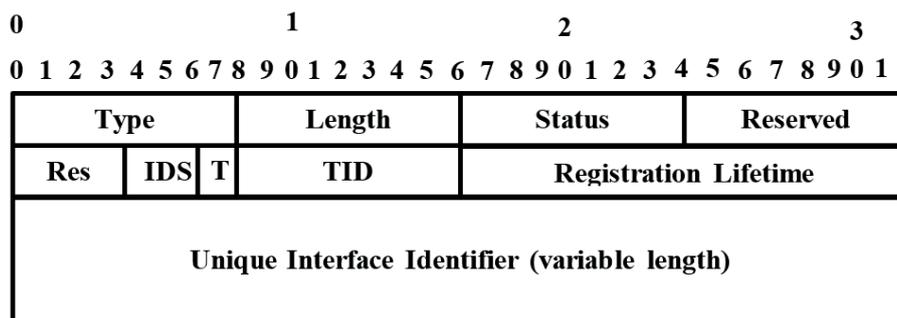
Fig. 20. The extensions to the fields in the ARO option [2]

Fields [2] *in the ARO:*

Type:	*33* [7]
Length:	*The length of the options in units of 8 bytes*
Status:	*8bit.This filed indicates the status of registration in the NA reply. This will be set to zero for NS messages.*
Reserved:	*8 bits. This field is unused.*
Res:	*4 bits. This field is unused.*
IDS:	*3 bits. Identifier name space. This specifies whether the UIID is DUID or EUI-64*
T bit:	*1 bit. Set f TID is valid.*
TID:	*8 bit. To find the most recent registration. Maintained by the EAH and used by NEAR.*
Registration Lifetime:	*16 bit. The amount of time the NEAR should maintain the NCE of the NS request received. A zero value will indicate that the entry is de-registering.*
Unique Interface Identifier:	*Variable length. Multiples of 8 bytes. Can be DUID or EUI-64*

The ARO option is included in the unicast NS, which contains the extensions to the ARO options in [7] like *Registration Lifetime*, *Unique Interface Identifier (UIID)*, *Transaction Identifier (TID)* and *SLLA*. The address registration process for an EAH is shown in Fig. 21. The NS with the ARO option is a request for the registration to the default NEAR by indicating its SLLA and the Registration Lifetime. The UIID and TID will be used by the NEAR to handle the DAD. The DAD check will detect whether the registration is to be approved or not. Once the ARO process is successful and DAD is cleared, the default NEAR will update its NCE with the registration by indicating the status, lifetime and registration approved time (to calculate the time left).The success message will be send in response through NA with the ARO status field to the EAH. The status field would indicate the success of registration or else an error message will be send in case of duplicate detection or cache full. In the case of a failure status, the tentative NCE created will be timed out and will not be present in the NCE. The NCE maintenance is further discussed in section 3.8.

REGISTERED NCE

Host	Status	Registration Lifetime(sec)
EAH	Incomplete	
HOST B	Stale	40
HOST C	Stale	15

Registration Request(NS)

ARO		SLLA
UIID	TID	
Registration Lifetime		

EAH

Default NEAR

REGISTRAR LIST

Registrar	Status	Registration Lifetime(sec)
Default NEAR	Reachable	1300
NEAR B	Stale	40

Registration Response(NA)

ARO Option

Status	0	1	2

Success Duplicate Cache full

REGISTERED NCE

Host	Status	Registration Lifetime(sec)
EAH	Reachable	1300
HOST B	Stale	40
HOST C	Stale	15

Fig. 21. Address Registration Procedure of EAH

To refresh the registration before its registration life time expires, the EAH need to send an NS with ARO to its default NEAR. The NEAR sends the NA with the re-registration reply along with updates (if any). If the host is sleeping the sleep time will be set in accordance to its registration life time so that it can wake up and refresh the registration before the lifetime expires to keep its address registration. While the EAH performs refresh by sending NS and receive NA, it handles movement detection and Neighbor Unreachability Detection (NUD) also.

Unlike RFC 4861 the registration entries are not garbage collected and hence when the lifetime expires they are explicitly unregistered. Also at any point of time EAH wishes to deregister from its NEAR it sends an NS with ARO by setting the Registration life time to zero. The overall operations that the optimization focuses on is shown in the signal flow graph in Fig. 22.
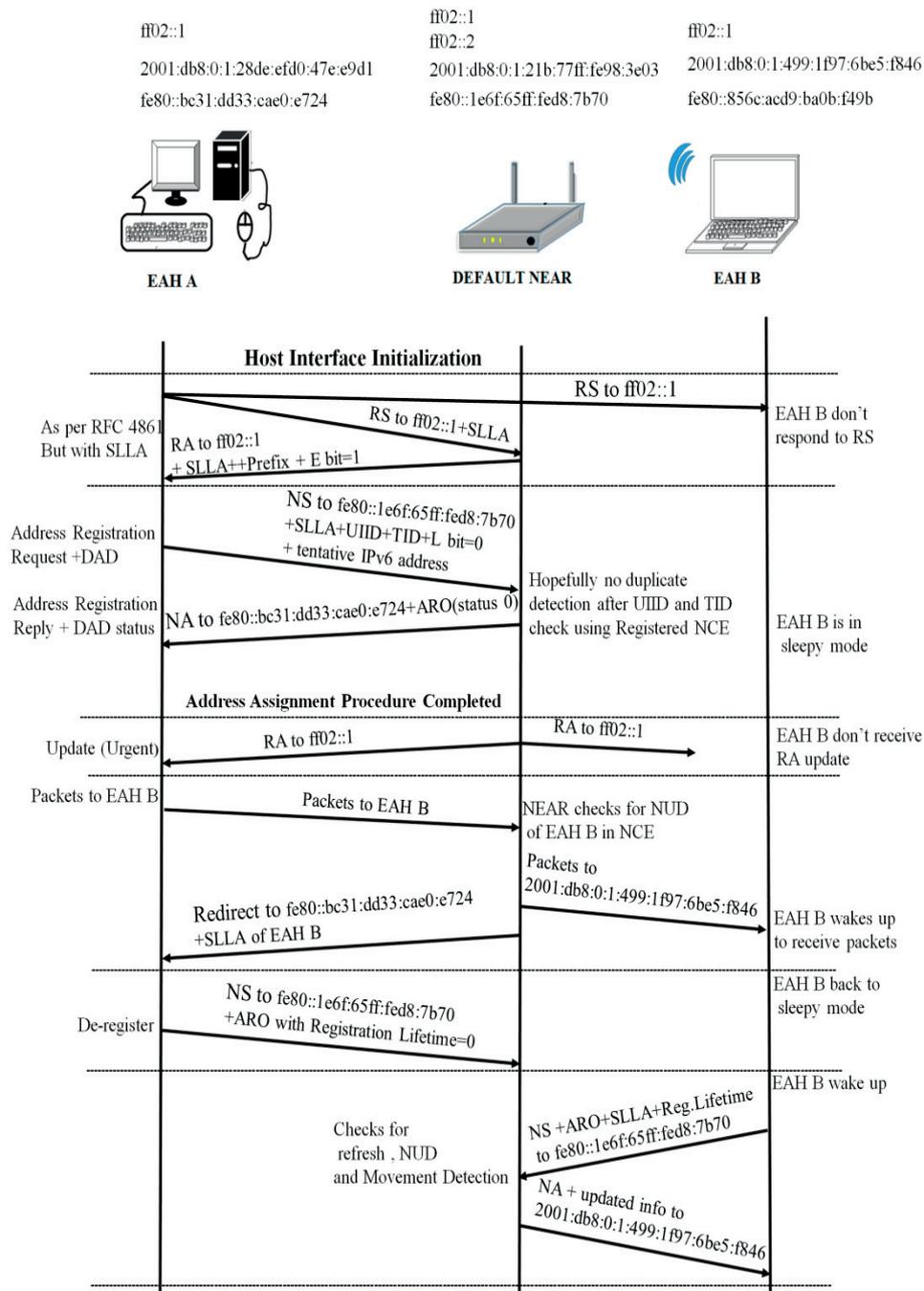
Fig. 22. A signal flow graph of the Address Registration, DAD, Address Assignment, and Refreshing of registration and De-registration procedures according to the efficiency –ND draft

3.7 Duplicate Address Detection Strategy

The unicast Neighbor Solicitation would perform the DAD according to the efficiency-ND draft. The ARO option has two identifiers to check for uniqueness of the tentative address requested for. The UIID and TID will be used by the NEAR to perform the DAD as shown in Fig. 23. The UIID will distinguish between an existing registration trying to refresh and a different host attempting to register an address that is already registered. The TID will be used by the default NEAR to find the most recent registration. If both UIID and TID makes the IPv6 address of the requested registration globally unique, then the DAD is cleared. The DAD clearance is not a severe problem in IPv6 due to the vast address field and hence will rarely have chances of duplicate detection.
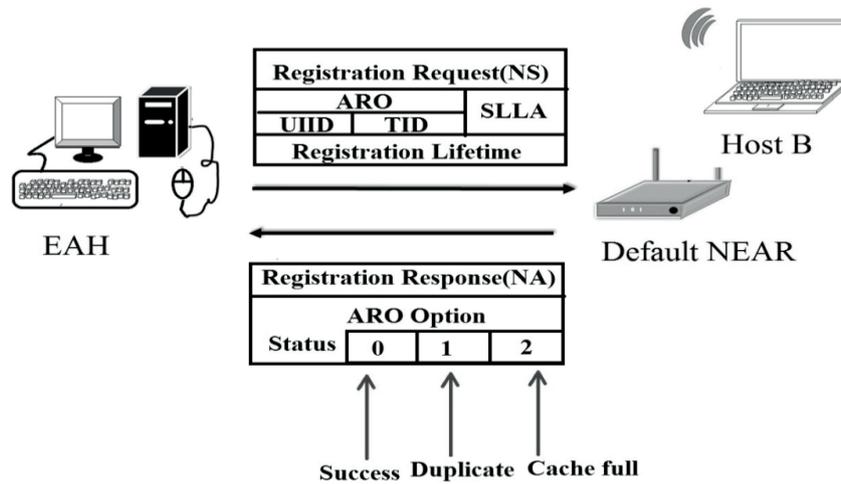
Fig. 23. The Duplicate Address Detection using unicast NS and ARO

3.8 Neighbor Cache Entry Maintenance

Unlike RFC 4861, as the draft makes the explicit address registrations with an associated registration life time and hence avoids multicast NS messages, the NCEs are handled differently in the draft to some extent. The entries are created as legacy entries first and hence are tentative type. On the successful handling of ARO, these tentative entries will be converted to Registered NCE entries. Until a NA is not forwarded from the NEAR the entry in NEAR NCE remains 'INCOMPLETE' entry. The entry will be created only if the NCE is not duplicate. When an NA is received in response to the NS sent, the sender updates the entry to 'REACHABLE'. In the NEAR, the registrations are made once DAD is cleared and then the

entry is updated to 'REACHABLE' state. When the life time is about to expire the entry is updated to 'STALE'. If the tentative NCE fails to register it gets deleted. If the NUD from the NEAR indicates the host is 'UNREACHABLE', the registration entry will be retained until the lifetime expires. The registered NCE will not be garbage collected whereas as per RFC 4861, the NCE will be garbage collected. Only one type of NCE can be maintained at a time. An example of registered NCE entry maintained in the NEAR is as shown in Table 2 and an example of NCE registration is shown in Fig. 24. Some of the key points adapted by NEAR to maintain the Registered NCE are as follows:

1. The NEAR will not make any modifications to the existing register entry for an EAH based on the RS.
2. If suppose an NS with an ARO option containing zero lifetime is encountered by the NEAR and cleared the DAD mechanism, it creates or updates the Registered NCE.
3. If the Neighbor Cache entry is full and it's impossible to make new entries, the NEAR sends a NA to the sender requested for registration with an ARO with status code set to 2.
4. If NEAR receives two NSs from the same EAH, one with ARO and one without ARO, the NS with ARO is attended.

TABLE 2. AN EXAMPLE OF REGISTRATION NEIGHBOR CACHE ENTRY MAINTAINED AT THE NEAR REGISTRAR

Host	IP address	MAC address	Status	Reg. Life time	Time Left (sec)
EAH A	2001:db8:0:1 :28de :efd0:47e:e9d1	1c:6f:65:d8:7b:70	Reachable	1800	1400
EAH B	2001:db8:0:1 :499 :1f97:6be5:f8eb	08:60:6e:7a:76:0a	Stale	1040	300
EAH C	2001:db8:0:1 :21b :77ff:fe98:3e03	60:a4:4c:52:5a:a5	Stale	1015	40

HOST INTERFACE INTIALIZATION

RS+SLLA

RS+SLLA

RA

NCE	
Neighbor	NCE state
Default NEAR	REACHABLE

Registered NCE	
Neighbor	NCE state
EAH B	REACHABLE

NS+ARO

NCE	
Neighbor	NCE state
Default NEAR	INCOMPLETE

Registered NCE	
Neighbor	NCE state
EAH A	INCOMPLETE
EAH B	REACHABLE

NA+ARO(Status=0)

NCE	
Neighbor	NCE state
Default NEAR	REACHABLE

Registered NCE	
Neighbor	NCE state
EAH A	REACHABLE
EAH B	REACHABLE

NCE REGISTRATION COMPLETED

EAH B entering sleepy mode

Registered NCE	
Neighbor	NCE state
EAH A	REACHABLE
EAH B	STALE

Packets to EAH B

Packets to EAH B

NS+ARO+SLLA

NCE	
Neighbor	NCE state
Default NEAR	REACHABLE
EAH B	INCOMPLETE

NCE	
Neighbor	NCE state
Default NEAR	STALE

NA+ARO(Status=0)

Registered NCE		
Neighbor	NCE state	NUD
EAH A	REACHABLE	
EAH B	STALE	REACHABLE

NCE	
Neighbor	NCE state
Default NEAR	REACHABLE
EAH A	REACHABLE via NEAR

Packets to
2001:db8:0:1:499:1f97:6be5:f846

Redirect to EAH A
+SLLA of EAH B

NCE	
Neighbor	NCE state
Default NEAR	REACHABLE
EAH B	REACHABLE via NEAR

Registered NCE	
Neighbor	NCE state
EAH A	REACHABLE
EAH B	STALE

EAH B going back
to sleepy mode

EAH B is waking up

NS+ARO+SLLA

Registered NCE	
Neighbor	NCE state
EAH A	STALE
EAH B	REACHABLE

NCE	
Neighbor	NCE state
Default NEAR	REACHABLE
EAH A	STALE

NA+ updates

NUD check for EAH A

EAH A is moved out
of the network

Registered NCE		
Neighbor	NCE state	NUD
EAH A	STALE (until lifetime expire)	UNREACHABLE
EAH B	STALE	

NS+ARO with lifetime=0

De-Register request

Registered NCE	
Neighbor	NCE state
EAH A	STALE (until lifetme expire)
EAH B	DELETE

EAH A   DEFAULT NEAR A   EAH B

Fig. 24. Neighbor Cache Entry at the NEAR and EAH

4 Implementation of RFC 4861 NDP

The preliminary stages of implementation procedure is the implementation of the legacy NDP and the capture of the packet flow. This stage is important to analyze how efficient the given draft is. The given draft will then be analyzed against the legacy NDP with a few test scenarios. These case scenarios will see if the draft is useful in case of transient wireless networks and helps scalability of the network.

4.1 Implementation Setup

The implementation of the legacy neighbor discovery protocol, according to [1], is performed using a simple desktop computer running on Ubuntu as a router. The host and the router need to be connected wirelessly on the same network of the Wi-Fi router. The Wi-Fi router used in the setup is basically a wireless switch. The router and the host ran the Ubuntu 14.04 version. The host is connected to the router using Wi-Fi. The Fig. 25 shows the implementation setup required for testing the legacy NDP.
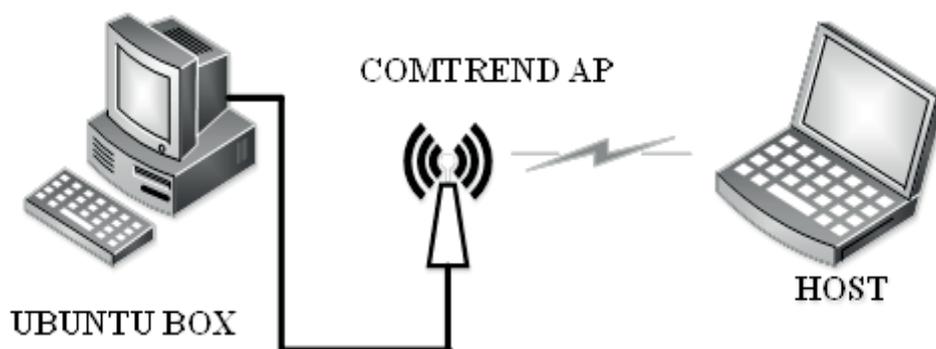
Fig. 25. The implementation setup for RFC 4861 NDP

4.2 Setting up the Wireless Router and Host

For the purpose of being able to replicate the experiment, a wireless router is created by connecting an Ubuntu PC to a wireless switch. As a primary

requirement, the router must route the packets from source to destinations and also send out RAs to the neighbors when they join the network.

To enable routing and packet forwarding, the kernel options in a few configuration files were changed. The system control configuration file in the *etc* folder is the command. On the command prompt */etc/sysctl.conf* file is edited to enable routing. The statement as shown in Fig.26 is included (or uncomment the statement if it already exists).

<div style="border:1px solid #000; text-align:center; padding:1em;">

net.ipv6.conf.default.forwarding=1

</div>

Fig. 26. Linux Command included in the sysctl.conf file to enable IPv6 forwarding

The other command that can be used directly on the terminal window instead of opening the config file is shown in Fig. 27.

<div style="border:1px solid #000; text-align:center; padding:1em;">

sysctl —w net.ipv6.conf.all.forwarding=1

</div>

Fig. 27. Alternate Linux command used to enable IPv6 forwarding directly on the terminal

Having enabled packet forwarding, the router also needs to send Router Advertisements. The open source software that is used to send out periodic RA is RADVD. RADVD sends out RAs that are in accordance to RFC 4861. Hence the software is installed. The installation commands are as shown in Fig. 28.

<div style="border:1px solid #000; text-align:center; padding:1em;">

apt-get install radvd

sudo vi /etc/radvd.conf

</div>

Fig. 28. Linux commands to install and configure radvd

The configuration file of RADVD is edited by adding the following settings as in Fig. 29.

```
interface eth0
{
        AdvSendAdvert on;
        prefix 2001:db1:0:1::/64
        {
                AdvOnLink on;
                AdvAutonomous on;
        };
};
```

Fig. 29. Code snippet: RADVD configuration settings

As code in Fig. 29 suggests, the network prefix is finalized here. Besides, the other flags like the M and O flags can also be set here. All this information is coded onto the router advertisement header.

The current interface of the router is configured with a static IP address that is a part of the network (this IP address will have the same network id as the advertised prefix). The commands for configuring the interface are given below. These commands go into the interfaces file in the network folder.

```
iface eth0 inte6 static
address 2001:db1:0:1::1
netmask 64
```

Fig. 30. Linux commands to edit the *interface* of the router

The interface is restarted to acquire a new id for the interface. This is given as shown in Fig. 31.

```
sudo /etc/init.d/networking restart
```

Fig. 31. Linux command to restart the interface of router

The routing information for this example is assumed to be static. The IP address is then added to the corresponding interface and the routes are added as well as shown in Fig. 32. This information gets into the *rc.local* file in the *etc* folder.

```
vi etc/rc.local

ip a a 2001:db1:0:1::1 dev eth1

ip r a 2001:db1:0:1::/64 dev eth1
```

Fig. 32. Linux command to add route to the configured interface of the router

Now the RADVD service is started using the command shown in Fig. 33.

```
service radvd start
```

Fig. 33. Linux command to start the radvd on the router

Now the Ubuntu pc has been modified to act like a router and send out RAs. When a new host is connected to its wireless network, they will receive RAs and they can be viewed by using the radvdump command. Ping6 command can be used to ping neighbors and see the flow of packets.

4.3 Experimental Output

The Wireshark used in the router as well as in the host makes it easy to see the sequence of packet flows on either directions and test the working of the [1]. Radvdump command can be used to see the arrival of the periodic RAs. The RAs appear every 200 seconds as per the draft. In order to make it easy to work with the system on various platforms the entire network is implemented on Virtual box environment. The results from this implementation are included below and interpreted as follows.

The screenshot in Fig. 34 shows Wireshark output of the legacy NDP on the router's side. The proper filters chosen can remove out the other unwanted packets from the Wireshark capture.

No.	Time	Source	Destination	Protocol	Length	Info
3	13.93	fe80::1e6f:65ff:fed8:7b70	ff02::1	ICMPv6	110	Router Advertisement from 1c:6f:65:d8:7b:70
7	29.93	fe80::1e6f:65ff:fed8:7b70	ff02::1	ICMPv6	110	Router Advertisement from 1c:6f:65:d8:7b:70
11	45.94	fe80::1e6f:65ff:fed8:7b70	ff02::1	ICMPv6	110	Router Advertisement from 1c:6f:65:d8:7b:70
17	55.61	::	ff02::1:ff98:3e03	ICMPv6	78	Neighbor Solicitation for fe80::21b:77ff:fe98:3e03
46	69.09	::	ff02::1:ff98:3e03	ICMPv6	78	Neighbor Solicitation for fe80::21b:77ff:fe98:3e03
55	70.09	fe80::21b:77ff:fe98:3e03	ff02::2	ICMPv6	70	Router Solicitation from 00:1b:77:98:3e:03
56	70.09	fe80::1e6f:65ff:fed8:7b70	ff02::1	ICMPv6	110	Router Advertisement from 1c:6f:65:d8:7b:70
58	70.12	::	ff02::1:ffe5:6d9b	ICMPv6	78	Neighbor Solicitation for 2001:db8:0:1:dd62:c395:80e5:
64	70.39	::	ff02::1:ff98:3e03	ICMPv6	78	Neighbor Solicitation for 2001:db8:0:1:21b:77ff:fe98:3
212	611.7	fe80::1e6f:65ff:fed8:7b70	ff02::1	ICMPv6	110	Router Advertisement from 1c:6f:65:d8:7b:70

```
                malicious@NEAR: ~
        RX bytes:6313 (6.3 KB)  TX bytes:6313 (6.3 KB)

malicious@NEAR:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 1c:6f:65:d8:7b:70
          inet addr:192.168.1.2  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: 2001:db8:0:1:95e1:43c0:f594:bc87/64 Scope:Global
          inet6 addr: fe80::1e6f:65ff:fed8:7b70/64 Scope:Link
          inet6 addr: 2001:db8:0:1:1e6f:65ff:fed8:7b70/64 Scope:Global
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:24745 errors:0 dropped:0 overruns:0 frame:0
          TX packets:789 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:14925845 (14.9 MB)  TX bytes:244266 (244.2 KB)
```

Fig. 34. Wireshark output of the legacy NDP on the router's side

The captured packets in Fig. 34 can be interpreted as following. The first three packets represent periodic RAs from the router to the 'all nodes multicast' group FF02::1. Somewhere between time 45 and 55 a new host enters the network. A NS packet is sent out by an unknown source (the source address is unknown because the new host does not have any address yet) to the solicited node multicast group ff02::1:ff98:3e03. This indicates that the new host either wants information about a particular host in the network or it wishes to register itself with a new address and is carrying on the process of DAD. It waits for 14 more time instants until it sends out another NS (though the actual protocol suggests only one NS this implementation has been made with two NSs).

In this case, the new host, is trying to register itself for a link local address of fe80::21b:77ff:fe98:3e03/64 based on the EUI 64 format from its MAC address 00:1b:77:98:3e:03.When no one else uses this address, the host configures itself with it, and begins to start looking for a global address. Hence it requests for a RA using a RS at time instant 70. Having received the router solicitation, the host now knows all information about the network. It choose an IP address and sends a NS for that address to the corresponding solicited node multicast group. From time instant 611, we only have periodic RAs flowing into the network. The same flow of packets can also be viewed upon the host computer's Wireshark screenshot as in Fig. 35.

49

Time	Source	Destination	Protocol	Info
0.33	::	ff02::1:ff98:3e0	ICMPv6	Neighbor Solicitation for fe80::21b:77ff:fe98:3e03
13.8	::	ff02::1:ff98:3e0	ICMPv6	Neighbor Solicitation for fe80::21b:77ff:fe98:3e03
14.8	fe80::21b:77ff:fff02::2		ICMPv6	Router Solicitation from 00:1b:77:98:3e:03
14.8	fe80::1e6f:65ff:ff02::1		ICMPv6	Router Advertisement from 1c:6f:65:d8:7b:70
14.8	::	ff02::1:ffe5:6d9	ICMPv6	Neighbor Solicitation for 2001:db8:0:1:dd62:c395:80e5:6d9b
15.1	::	ff02::1:ff98:3e0	ICMPv6	Neighbor Solicitation for 2001:db8:0:1:21b:77ff:fe98:3e03
556.	fe80::1e6f:65ff:ff02::1		ICMPv6	Router Advertisement from 1c:6f:65:d8:7b:70
829.	fe80::1e6f:65ff:ff02::1		ICMPv6	Router Advertisement from 1c:6f:65:d8:7b:70

Fig. 35. Wireshark output on the host side

Some data is exchanged between the host and the router by using the ping6 functionality. The host first sent out NS for resolving the router's address and received an NA from the router followed by several data packets (in this case ping request and replies) as shown in Fig. 36.

Time	Source	Destination	Protocol	Info
611.	fe80::1e6f:65ff:fed8:7b70	ff02::1	ICMPv6	Router Advertisement from 1c:6f:65:d8:7b:70
885.	fe80::1e6f:65ff:fed8:7b70	ff02::1	ICMPv6	Router Advertisement from 1c:6f:65:d8:7b:70
1202.	fe80::1e6f:65ff:fed8:7b70	ff02::1	ICMPv6	Router Advertisement from 1c:6f:65:d8:7b:70
1288.	2001:db8:0:1:dd62:c395:80	ff02::1:ff94:bc87	ICMPv6	Neighbor Solicitation for 2001:db8:0:1:95e1:43c0:f594:bc87 from
1288.	2001:db8:0:1:95e1:43c0:f5	2001:db8:0:1:dd62:	ICMPv6	Neighbor Advertisement 2001:db8:0:1:95e1:43c0:f594:bc87 (rtr, sc
1288.	2001:db8:0:1:dd62:c395:80	2001:db8:0:1:95e1:	ICMPv6	Echo (ping) request id=0x0e1c, seq=1, hop limit=64 (reply in 319
1288.	2001:db8:0:1:95e1:43c0:f5	2001:db8:0:1:dd62:	ICMPv6	Echo (ping) reply id=0x0e1c, seq=1, hop limit=64 (request in 318
1289.	2001:db8:0:1:dd62:c395:80	2001:db8:0:1:95e1:	ICMPv6	Echo (ping) request id=0x0e1c, seq=2, hop limit=64
1289.	2001:db8:0:1:95e1:43c0:f5	2001:db8:0:1:dd62:	ICMPv6	Echo (ping) reply id=0x0e1c, seq=2, hop limit=64 (request in 326
1290.	2001:db8:0:1:dd62:c395:80	2001:db8:0:1:95e1:	ICMPv6	Echo (ping) request id=0x0e1c, seq=3, hop limit=64
1290.	2001:db8:0:1:95e1:43c0:f5	2001:db8:0:1:dd62:	ICMPv6	Echo (ping) reply id=0x0e1c, seq=3, hop limit=64 (request in 328
1291.	2001:db8:0:1:dd62:c395:80	2001:db8:0:1:95e1:	ICMPv6	Echo (ping) request id=0x0e1c, seq=4, hop limit=64
1291.	2001:db8:0:1:95e1:43c0:f5	2001:db8:0:1:dd62:	ICMPv6	Echo (ping) reply id=0x0e1c, seq=4, hop limit=64 (request in 336
1292.	2001:db8:0:1:dd62:c395:80	2001:db8:0:1:95e1:	ICMPv6	Echo (ping) request id=0x0e1c, seq=5, hop limit=64 (reply in 333
1292.	2001:db8:0:1:95e1:43c0:f5	2001:db8:0:1:dd62:	ICMPv6	Echo (ping) reply id=0x0e1c, seq=5, hop limit=64 (request in 332
1293.	2001:db8:0:1:dd62:c395:80	2001:db8:0:1:95e1:	ICMPv6	Echo (ping) request id=0x0e1c, seq=6, hop limit=64 (reply in 336
1293.	2001:db8:0:1:95e1:43c0:f5	2001:db8:0:1:dd62:	ICMPv6	Echo (ping) reply id=0x0e1c, seq=6, hop limit=64 (request in 335

Fig. 36. Wireshark output on the routers side when pinged from the host

CHAPTER 5

5 State Changes of Routers and Hosts : A study

The implementation of the draft's modified neighbor discovery protocol is achieved by primarily changing the router to a NEAR. This involves packet format changes of the ICMPv6 messages and including a few new source files. This is then followed by changing the hosts to an EAH. Since the whole source code is a part of the Linux kernel, changes must be made and a new kernel must be compiled that will run in the host computers. But complete changes to the kernel is not performed in this thesis work. This chapter will clearly explain the design made for implementing the code changes in addition to all the changes implemented in this work.

5.1 State change of router to NEAR

The principal changes with the router includes letting the hosts know that, it is not a legacy router and it works in the energy efficient mode. Hence the RA will have to carry this information. The draft suggests inclusion of a new E flag in the RAs. This will be a part of the 6 reserved flags.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Cur Hop Limit |M|O|  Reserved |         Router Lifetime       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Reachable Time                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Retrans Timer                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Options ..
+-+-+-+-+-+-+-+-+-+-+-

    0 1 2 3 4 5 6 7
   +-+-+-+-+-+-+-+-+
   |M|O|H|Prf|P|E|R|
   +-+-+-+-+-+-+-+-+
```
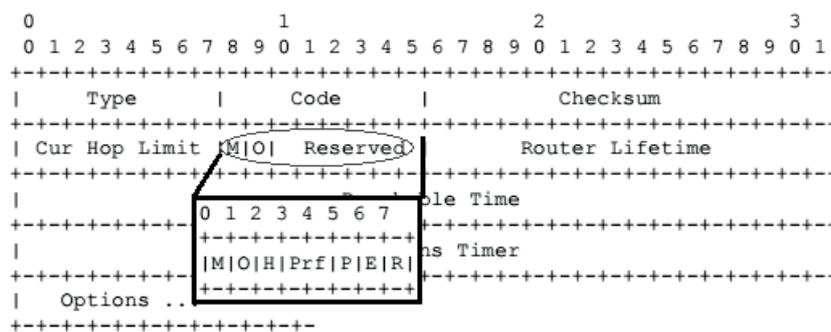
Fig. 37. Inclusion of new flags in RAs as per the draft

51

Though the draft suggests 6 flags as shown in Fig. 37, only the E flag is required for this implementation and hence was implemented. Since RADVD is the open source software used to make a host as a router, code changes are made in the RADVD source code. The source code for RADVD from GitHub was forked and downloaded and the changes were made. The header of the router advertisement contains all the flag information. Hence the structure in *radvd.h* file was changed as shown in Fig. 38.

```
struct ra_header_info {
        int AdvManagedFlag;        ←——— int AdvEFlag;
        int AdvOtherConfigFlag;
        uint8_t AdvCurHopLimit;
        int AdvHomeAgentFlag;
        int32_t AdvDefaultLifetime;
        int AdvDefaultPreference;
        uint32_t AdvReachableTime;
        uint32_t AdvRetransTimer;
} ra_header_info;
```

Fig. 38. Modification of the RA packet

The *send.c* files contain the functions which includes creation of the packet and sending it to the hosts over a particular interface. The function *add_ra_header* is responsible for this task.

Another change would be to include the E flag bit as a macro for programming readability purpose in the *icmp6.h* file. A choice of *0x10* is made for the E flag and hence choosing the *fourth bit* as the E flag bit. To increase ease of working using the *radvd.conf* file the corresponding *radvd.config.5.man* file can be changed as shown in Fig. 39 and entries for the E field can be created. Using this file, the user can now set the E field and the router can now act as a NEAR. The default value for the E flag was also set in the *defaults.h* as a macro. The default value is generally *0* but in this case, because mixed mode case or the legacy case is not implemented, the default value is set as *1* (all the routers in our experiment will be NEARs and no other mode of operations are tested for this implementation). The corresponding changes were also made in the *radvdump.c* file. The *radvdump* is the command that is used to view an incoming RA.

52

```
.TP
.BR AdvManagedFlag " " on | off

When set, hosts use the administered (stateful) protocol for address
autoconfiguration in addition to any addresses autoconfigured using
stateless address autoconfiguration.  The use of this flag is
described in RFC 4862.

Default: off                                      AdvEFlag information
                                            ←――――――――――――――

.TP
.BR AdvOtherConfigFlag " " on | off

When set, hosts use the administered (stateful) protocol for
autoconfiguration of other (non-address) information.  The use of
this flag is described in RFC 4862.
```

Fig. 39. Modification of the radvd.conf file

Besides the changes in the RA, the hosts will also be registering themselves to the default router. Hence the router must have a data structure that constantly stores, retrieves and updates information about all the hosts that are attached to it. The easiest form of working on host data is to use a hash table. That information must be programmed in the router's kernel because RADVD will only be used for sending Router Advertisements. This task is not included in this work, but a simple design of the router's hash table looked as shown in Table 3.

TABLE 3. AN EXAMPLE OF A NEAR'S NCE

	IP address	MAC address	Registration time period
Host A	2001:db1::44:2	33:6F:42:18	352
Host B	2001:db1::2463:325:4	9A:BC:12:74	41

Apart from the RA, the NA and the NS will also undergo packet format changes. The SLLA field, as explained in chapter 3 section 3.4 will become compulsory in all packets. Also the optional fields should include the ARO options in both the NA and NS. The NEAR will send out NSs instead of the hosts, as explained earlier. All the information about neighbor discovery besides sending RAs lie in the *ndisc.c* file in the *net/ipv6* folder of the Linux GitHub code [5].

A new subroutine called *ndisc_create_aro* has to be created which creates the ARO options. This ARO option is appended to the NA and the NS

packets. The functions in *line 477* and *line 554* of the *ndisc.c* will call the *ndisc_create_aro* function.

As per the draft, the NEAR has to send NA on behalf of all hosts that are registered to itself. When it receives a NA for a host to which it is a default router, it need to send out unicast NA. Hence the function *ndisc_recv_na* in *line 862* should be modified accordingly. All NSs and NAs will be unicast and no longer be sent to solicited node multicast group.

5.2 State change of router to NEAR

The major change when it comes to changing a regular host to an EAH is to decode the RAs and identify the E flag set by the NEARs. While sending out NSs during DAD and to send packets to another host, the EAH sends packet to its default router and not to its solicited node multicast group. The identification of the E flag is performed in the *ndisc.c* file in the *ndisc_router_discovery* function as shown in Fig. 40. The *if_flags* are 32 bit unsigned integers and so will the E flag be too.

The value *0x08* is chosen as for the E flag. This value was used in the *if_inet6.h* file in *line 26*. A macro is included in *line 75* of *linux/icmpv6.h* file for the E flag and included the E flag field in the *lines 43 and 50*. Hence the E flag is now decoded and the legacy host starts working as an EAH.

```
*/
in6_dev->if_flags = (in6_dev->if_flags & ~(IF_RA_MANAGED |
                     IF_RA_OTHERCONF)) |
                    (ra_msg->icmph.icmp6_addrconf_managed ?
                        IF_RA_MANAGED : 0) |
                    (ra_msg->icmph.icmp6_addrconf_other ?
                        IF_RA_OTHERCONF : 0);          ⟵  IF_E_FLAG

if (!in6_dev->cnf.accept_ra_defrtr) {
        ND_PRINTK(2, info,
                "RA: %s, defrtr is false for dev: %s\n",
                __func__, skb->dev->name);
        goto skip_defrtr;
}
```

Fig. 40. Reading the E flag on the EAH from the RAs

To complete the working of an EAH, it must stop sending NSs to solicited node multicast group and send all requests to its default router. This change is implemented in *line 554* of *ndisc.c* under the sub routine *ndisc_send_ns*. Also while sending NAs the EAH will advertise only to its default router using unicast. This is changed in the *ndisc_send_na* of the same source file.

CHAPTER **6**

6 Analysis of Efficient Draft-ND

This chapter makes an empirical test analysis of different IPv6 ND test case scenarios for the legacy NDP in a general wireless network in terms of the scalability and transient nature of the network. The results of the empirical analysis is used to make a theoretical analysis of the efficient-ND draft. The experimental set up used for the empirical analysis is shown in Fig. 41.
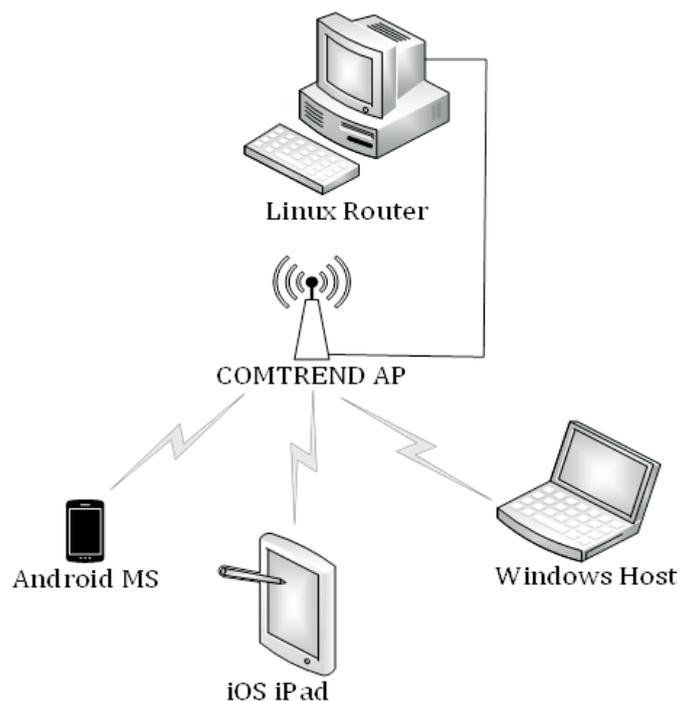
Fig. 41. Implementation setup of the network for the empirical analysis

A Linux machine is used as the router in which Ubuntu 15.04 version is running which is different from the implementation set up used in Chapter 4. This Ubuntu box will be configured to act as an IPv6 router. In order to

provide the ND, the router should be able to send router advertisements as mentioned also in chapter 4. The open source software package Router Advertisement Daemon (radvd) 1.9.1 version is the latest version in Ubuntu 15.04 that supports the router to listen to the router solicitations and to send router advertisements. This package is used in the Linux router. Most of the implementation procedures are addressed in chapter 4. But few steps are repeated in this chapter for the continuity of the details during the analysis.

6.1 Router Configuration

A brief look at the configurations made in the router is shown in Fig. 42. The main five steps shown would set the Ubuntu machine to act as a router.
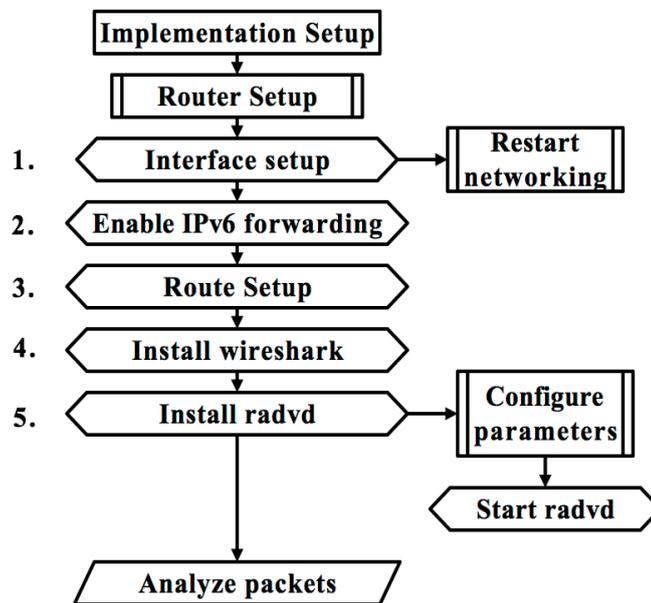
Fig. 42. Set up procedure at the router to analyze the packets in the network

The router configuration steps are shown in details in the Appendix 1. Multiple test case scenarios are considered on these three wireless devices to display or estimate the similarities and dissimilarities in their behavior during the IPv6 ND implementations. The behavior of these devices must be analyzed to check the scalability and transient nature of the network. The basic ND behavior of a Canon Wi-Fi enabled Printer is also analyzed to represent an M2M scenario while implementations and comparisons for this device is not considered.

This chapter further discusses the different test case scenarios along with their expected behavior according to the legacy NDP and also takes a glance through the expected behavior in accordance with the efficient-ND draft during these scenarios. The expected results would be analyzed in the following chapter. This chapter would be concluded with a brief discussion of the various parameters considered for the empirical analysis of the legacy NDP and theoretical analysis of the draft.

6.2 Test case scenarios

Some of the test case scenarios are analyzed for legacy NDP and efficient–NDP. The analysis is made mainly in relation to the messages exchanged between the devices and also with the router. Hence the test cases are mainly categorized as:

- IPv6 ND Behavior
- Scalability of the network
- Transient Behavior of the network
- Multicast messages exchanged in the network
- Sleep and wake up scenario

6.2.1 IPv6 ND Behavior

When a system boots up, as stated in [1] section 6.3.7, it transmits multicast RS to the all-routers multicast address. As a reply, all the routers in the network multicast RAs to the all-nodes multicast group address. According to the network scenario under consideration, there exist only one router that reply to RS. The IPv6 address configuration on the interface would be in respect to the specific IPv6 implementations on the wireless device. These IPv6 addresses can be global addresses or link local addresses or both. The global addresses can be either privacy addresses, EUI-64 based address or both, whereas the link local address can be either EUI-64 format address or random Interface ID addresses. Hence one NS each is sent for each of these addresses and a NA is expected for each duplicate address detected.

Before address binding, the node needs to join a solicited node multicast group. As stated in [1] section 7.2.1, in order to join this solicited node multicast group, the node sends Multicast Listener Report (MLDv2) [10] on

the link to find the presence of other multicast listeners on the link it is attached to.

Correspondingly for draft-ND, when the system boots up, the case is similar to the legacy ND and thus it multicasts RS to the all-router multicast group after joining the all-nodes multicast group. As stated in [2], the SLLA option is included in the RS for the routers on the link to reply with unicast RA. When the node receives more than one unicast RA from different routers, the node makes its default router selection as mentioned in [1] section 6.3.6. In the implementation setup on which this project work makes analysis, only one router is deployed and therefore it is the default router.

Now for the address binding, if the node uses a EUI-64 format to form its link local address, the DAD is not needed to be performed as they are unique and formed from the hardware address. Similarly during the global address binding, as seen in [7] section 3.1, if the EUI-64 format is used to form the IPv6 address, a DAD check is not required as they are globally unique. Meanwhile if the global IPv6 addresses are privacy addresses, they need to be checked for duplicate detection since they are not assured to be globally unique.

Whenever a link local address and/or global IPv6 address is formed apart from the EUI-64 format, 1 NS with the ARO option is send for each of those privacy addresses or random interface ID addresses. A NA message is expected for this NS with the ARO status of success or failure. Based on the efficient-ND draft [2] section 8.1, the node does not need to join the solicited node multicast group. And thus no Multicast Listener Report messages are exchanged.

6.2.2 Scalability of the network

Analyzing this scenario would evaluate the effect of growing number of nodes within the network and also the effects of this growth on the nodes already joined the network.

In case of legacy NDP, when more nodes join the network, the ND behavior of each node joined would have effect on every other node already joined the network. Considering [1], most of the messages are multicast and thus these messages would reach other nodes attached to the network and would be inspected by them even if they are not relevant to them.

Consequently the number of multicast messages within the network increases with the number of nodes. The current wireless network includes devices that goes to sleep to save their battery life. This sleep and wake up approach is not addressed in [1] NDP. Therefore the effect of multicast messages on the node will be according to the power saving techniques adopted by the node. This situation is discussed in section 6.2.5.

Now for the efficient draft-ND, periodic RAs are completely avoided and most of the multicast messages are avoided as well. As a result only the RS messages sent during connection setup are mainly the only multicast messages. The number of these RS messages sent for each node depends on its manufacturing configuration. In standard situations, each node during system boot up send a RS and hence the number of multicast messages increases in the network as a multiple of the number of nodes within the network. This can vary slightly depending on the device manufacturing details. In the efficient-ND, if a node is in its sleep cycle, it does not accept any messages and furthermore there are no chances of any multicast messages other than the occasional emergency RA updates. The analysis for the sleep state of the node is out of the scope of this thesis work. Any discussions made in relation to the sleep and wake up scenario in this work is a theoretical perspective rather than an implementation point of view.

6.2.3 Transient Behavior of the network

The current wireless networks has a vast number of wireless devices and the network never remains the same for a long time. Nodes join and leave the network frequently and this makes the nature of the network impermanent or transient. This case should be tested as the flow of multicast massages within the network during this situation is highly relevant.

In the legacy ND, nodes send Multicast Listener Report and Multicast Listener Done message on an attached link while leaving a network as indicated in [9] and in its updated version [10]. This behavior can vary depending on whether the silent leave (soft leave) mechanism is enabled in the wireless device or a fast leave processing. In the soft leave processing, the node can ignore the periodic MLD messages and leave silently or it can send an *MLD filter mode change record.* In contrast, for the fast leave processing, the nodes send MLD messages to block the old sources for a specific address group (*BLOCK_OLD_SOURCES*). These details are

explained elaborately in section 5.2.12 of [10]. The number of MLDv2 messages exchanged would vary depending on the implementations.

The transient nature of the network can be due to:

- Link Failure
- Link Re-establishment
- Node leaving the network

Link failure can be either due to unstable Wi-Fi or other technical failures and Link re-establishment happens when the Wi-Fi becomes stable. As stated in section 6.2.1, efficient ND draft does not support Multicast Listener messages as there are no members in the solicited node multicast group. Hence once the node joins the network and registers itself with its default registrar, the case becomes slightly easier until its registration lifetime expires.

If the link failure is due to the unstable Wi-Fi or movement, the node will not send any message, instead the node can re-establish its connection with the router with an initial unicast NS with ARO option to its default router using its NCE. This will help to refresh its address registry. No multicast messages are involved here unless the registration lifetime is expired. However if the registration lifetime is expired, the connection must be established from the start to find its default registrar.

In the case of a node leaving the network, it should send a de-registration message to its default router (registrar). This action is carried out by sending a unicast NS with the ARO option where its registration lifetime is set to zero.

6.2.4 Multicast messages exchanged in the network

As explained in [1] section 6.2.1, the router is configured to have certain variables on each interface that is placed in its outgoing RA messages. Host nodes uses these information in the RA to control their external behavior. Some of these variables can be altered to control the frequency of the RA messages. The most important variables among these are the *AdvDefaultLifetime*, *MinRtrAdvInterval* and *MaxRtrAdvInterval*. These variables are discussed more in detail in the upcoming section 6.3.

In this test case scenario, the number of multicast RA messages and the total number of multicast messages exchanged in the network can be investigated using an implementation aspect for the legacy ND and a theoretical aspect for the efficient ND draft. The above mentioned router variables can be varied for different time durations and a comparison of multicast messages exchanged in both the ND cases can be analyzed theoretically.

6.2.5 Sleep and wake up scenario

This case scenario is merely discussed and not analyzed in this thesis work. The efficient ND addresses this scenario while the legacy ND does not attend this in particular. The significant growth of the current mobile networks requires need for efficient usage of battery power as most of them are low power communication devices. They often communicate frequently.

Merely restricting these devices from the unnecessary transmissions would not be enough to achieve a better power consumption. Having a better power saving technique is the key problem to be solved. Various power saving methods [11] [12] [13] are preferred by the manufacturers for different devices to make their product an energy efficient device. Since there is no standardized power saving technique applied by all the vendors, the wireless devices exhibit different behaviors in its sleep and wake up cycle.

Duty cycling techniques such as Always ON, Normally OFF, and Low-power were proposed [14] for wireless devices depending on the frequency of communication on these devices. Most of the modern wireless devices makes use of these varied duty cycles to reduce the power consumption. The radio systems of the clients consume power even when the mobile is in the sleep mode. Some of the hosts lets itself to listen to the multicast messages in the network. This would eventually be helpful for the DAD [15] in the modern wireless networks though this process consumes a huge amount of energy unnecessarily.

On contrary, some devices are constructed with power saving mechanisms that completely switches off the radio and goes to sleep cycle. These hosts can save their battery life slightly better although sacrifices their addresses [15] by not listening to the messages during their sleep cycle. Additionally, the energy incurred in forming new addresses as well as the risk of the

communications addressed to these devices being directed to some other devices are also associated issues. Therefore when the device wakes up, there exist one of the two possible following situations [15]:

- Host has a unique IPv6 address
- Host has a duplicate IPv6 address.

These vital problems are addressed in the efficient ND draft by making a registration with the default router thereby avoiding any periodic RA and most of the multicast messages. Despite the fact that different wireless devices functions divergently to achieve the power saving mechanism, these techniques would not affect the neighbor discovery as per the NDP mentioned in the efficient ND draft. Resultantly the host devices can complete their sleep cycle without any disturbances.

6.3 Parameters considered for the theoretical analysis

As indicated in section 6.2.4, the router variables that can control the frequency of the periodic multicast RA messages on a link are:

- AdvDefaultLifetime
- MaxRtrAdvInterval
- MinRtrAdvInterval

6.3.1 AdvDefaultLifetime

This is the lifetime of the associated default router on the link chosen by the node. In [1] section 4.2, this variable is mentioned to have a value up to *65535 seconds*, whereas for the implementations the value was limited to *9000 seconds*. Whenever the variable has a value *zero*, the router is no longer a default router.

6.3.2 MaxRtrAdvInterval

This is the maximum time in seconds, allowed between sending unsolicited multicast RA messages from the interface. This takes a value no less than *4 seconds* and no greater than *1800 seconds* as specified in section 6.2.1 of [1]. Later this value is updated to a maximum allowed value to avoid unnecessary unsolicited RAs. This maximum limit is intended to be *21845 seconds* for the current NDP scenarios [16].

6.3.3 MinRtrAdvInterval

This is the minimum time in seconds, allowed between sending unsolicited multicast RA messages from the interface [1]. This takes a value no less than *3 seconds* and no greater than *0.75*MaxRtrAdvInterval*.

Extensions to update [1] to improve the usage of IPv6 ND to save the battery life led to updates of these values as mentioned in the latest version of [16]. According to the update in this draft, the relationship between the *AdvDefaultLifetime* and the *MaxRtrAdvInterval* is the ratio shown in (1)

$$k=AdvDefaultLifetime/MaxRtrAdvInterval \tag{1}$$

The *k* factor can show how many RAs will be sent in a network by the default router before its router lifetime expires.

In accordance with [16], *AdvDefaultLifetime* is updated to a maximum value of *65535 seconds* and the value ranges from *MaxRtrAdvInterval* to *65535 seconds*. The *MaxRtrAdvInterval* will have an updated value that is not greater than *21845 seconds* for the standard implementations. Thus these updates on [1] is currently implemented in the current wireless device ND to reduce the frequency of periodic RAs in the network thereby improving the efficiency.

Even with these updates, the periodic RAs are not completely avoided as well as the other multicast messages. However the efficient ND approach completely avoids the periodic unsolicited RAs and also most of the multicast messages except the initial RS messages send during the connection setup.

CHAPTER **7**

7 Expected Performance Results

This chapter describes the results from different test cases mentioned in chapter 6. These observations are analyzed with the expected behavior for the efficient-ND draft. An analysis for these results are graphically presented for a better understanding. It is important to note that the observations for the test case scenarios analyzed here even though are correct for basic evaluation of the ND behavior, they can vary from implementation to implementation.

7.1 Test case scenarios

The test case scenarios discussed in the chapter 6 section 6.2, will be analyzed below with its observed results for legacy ND and expected results for efficient draft.

7.1.1 IPv6 ND Behavior

The neighbor discovery messages exchanged during the system boot up in the cases of an Android MS, iOS iPad and a Windows Host is shown in Fig. 43, Fig. 44 and Fig. 45. Note that these figures shown are for the implementations in the current wireless network that is based on the [1]. The comparisons are mainly done for the multicast messages exchanged. The Android device on system boot up multicasts 1 RS, whereas iOS iPad sends 2 RS and the windows sends 1 RS which is shown in Fig. 51. The maximum number of solicitations transmitted in order to obtain RA immediately will be *3 transmissions (MAX_RTR_SOLICITATIONS)* each separated by *4(RTR_SOLICIATION INTERVAL) seconds*. The routers send multicast router advertisements in response to these solicitations. Then it does address binding by checking DAD and also sends multicast listener reports.
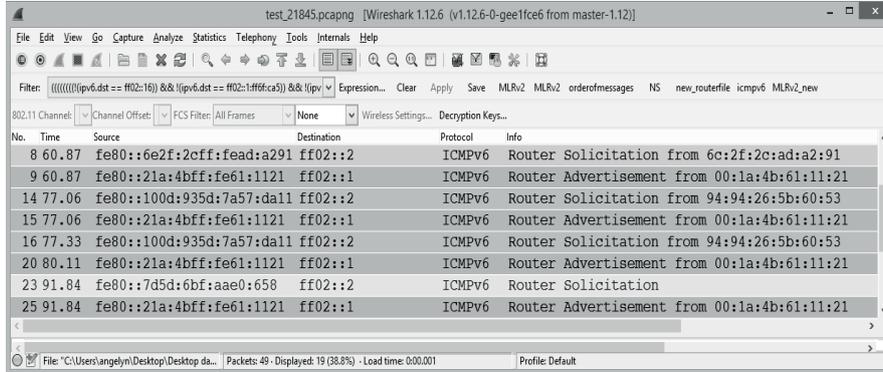
Fig. 43. RS/RA Messages at the system startup of Android MS, iOS iPad and Windows host
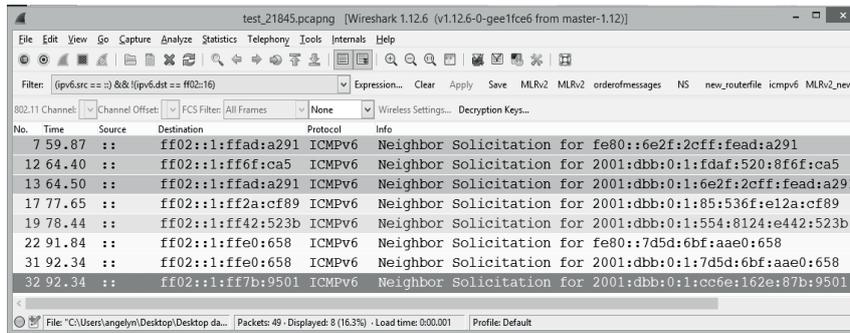
Fig. 44. Duplicate Address Detection for Android MS, iOS and Windows host
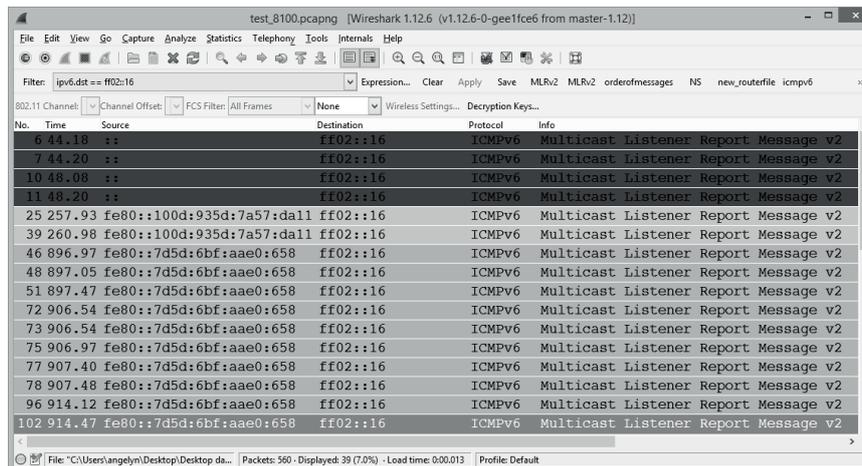
Fig. 45. Multicast Listener Report v2 for Android MS, iOS and Windows host

The specific behaviors of the devices for normal implementations while connection set up would be:

i. Android MS sends 1 RS to the all-router multicast group ff02::2. The router on link sends RA as multicast to the all-node multicast group ff02::1. It generates one link local address and two IPv6 global addresses. The link local address *fe80:: 6e2f:2cff:fead:a291* is formed from the EUI-64 format that uses the hardware address (*6c:2f:2c: ad: a2:91*). One of the global address (*2001: dbb: 0:1:6e2f:2cff: fead: a291*) is EUI-64 based and the other one is a privacy address (*2001: dbb: 0:1: fdaf: 520: 8f6f: ca5*). TheIPv6 addresses use the prefix advertised by the default router in its RA which is *2001: dbb: 0:1* in the case of the implementation conducted. It performs DAD for all these addresses by sending neighbor solicitations to the solicited node multicast group as shown in Fig. 44. The MS sends 4 Multicast Listener Report messages (MLDv2) as shown in Fig. 45. Thus a total of 9 multicast messages are exchanged at connection setup.

ii. The iOS iPad sends 2 RS to ff02::2 to which the router on link sends 1 RA each as multicast to ff02::1. It generates one link local address and two IPv6 global addresses. The link local address is formed from a random interface ID. One of the global address (*2001: dbb: 0:1:554:8124: e442: 523b*) is based on a random IID and the other one is a privacy address (*2001: dbb: 0:1: 85: 536f: e12a: cf89*) which is the same for all the implementations. It performs DAD only for the global addresses and not for the link local address as shown in Fig. 44. The iOS device sends 2 Multicast Listener Report messages (MLDv2) as shown in Fig. 45. Thus a total of 8 multicast messages are exchanged at connection setup.

iii. The Windows host sends 1 RS to which the router on link sends RA as multicast. It generates one link local address and two IPv6 global addresses. The link local address is formed from a random interface ID (*fe80:: 7d5d:6bf:aae0:658*). One of the global address (*2001: dbb: 0:1: 7d5d: 6bf:aae0:658*) is based on the last 64 bits of the link local address and the other one (*2001: dbb: 0:1: cc6e: 162e: 87b: 9501*) is a random IID based. It performs DAD for all these addresses as shown in Fig. 44. The Windows host sends 11 Multicast Listener Report messages (MLDv2) as shown in Fig. 45. Thus a total of 16 multicast messages are exchanged at connection setup.

The difference in the total number of multicast messages for the different devices is mainly due to the dissimilarity in the number of DAD messages and the MLDv2 messages.

Now the theoretical results in case of the efficient ND modified protocol can be analyzed in relation to the above implemented results. The expected behaviors of the devices would be:

i. Android MS sends 1 RS to the all-router multicast group *ff02::2* but includes *SLLA* option. The router on link sends RA as unicast to the *SLLA* of the MS. It generates one link local address and one global IPv6 address from the EUI-64 format that uses the hardware address and another global IPv6 address which is a privacy address. It performs DAD for only the privacy address as EUI-64 based addresses does not require DAD check. The DAD is performed by sending a unicast neighbor solicitation with ARO option to the router. A NA message is received if the DAD check is a success. The MS will not send any MLDv2 messages. Thus out of the total 4 messages exchanged at connection setup, only 1 message is multicast.

ii. The iOS iPad sends 2 RS with SLLA option included to *ff02::2* to which the router on link sends 1 RA each. The router on link sends RA as unicast to the *SLLA* of iOS device. It generates one link local address from a random interface ID and two IPv6 global addresses, one based on a random IID and the other one a privacy address. It performs DAD only for the two global addresses and not for the link local address. This DAD check will receive 2 NA messages in return to indicate the success or failure of the process. Note that both these 2NS/NA messages are unicast messages. The iOS device will not send any MLDv2 messages. Thus out of the total 8 messages exchanged at connection setup, only 2 messages are multicast.

iii. The Windows host sends 1 RS that includes *SLLA* option to the *ff02::2* to which the router on link sends 1 unicast RA. It generates one link local address from a random interface ID and two IPv6 global addresses, one of them based on the last 64 bits of the link local address and the other one based on a random IID. It performs DAD for all these addresses. The Windows host does not send any MLDv2 messages. Thus out of the total 8 messages exchanged at connection setup, only 1 message is multicast.

A theoretical comparison of the results seen above is shown in Fig. 46.
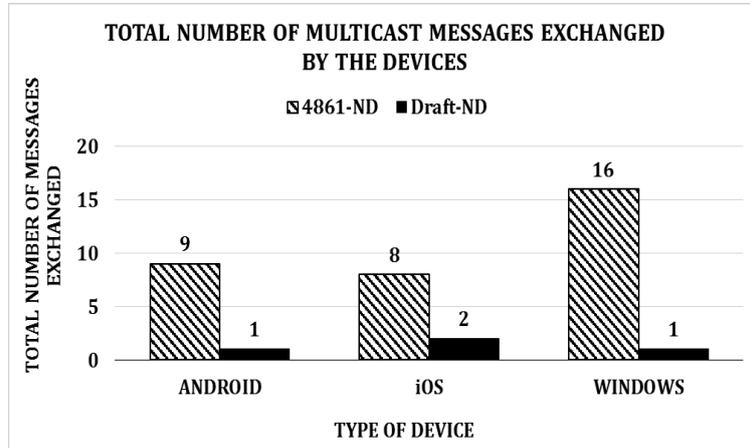
Fig. 46. Total multicast messages exchanged during the connection setup for Android MS, iOS IPad and Windows host

The total messages exchanged irrespective of multicast or unicast is shown in Table 4 and Fig.47.

TABLE 4. A COMPARISON OF TOTAL MESSAGES EXCHANGED AT THE CONNECTION SETUP FOR DIFFERENT SYSTEMS.

Types of Messages Exchanged	Android MS		iOS		Windows		Canon	
	4861-ND	Draft-ND	4861-ND	Draft-ND	4861-ND	Draft-ND	4861-ND	Draft-ND
RS	1	1	2	2	1	1	1	1
RA	1	1	2	2	1	1	1	1
NS	3	1	2	2	3	3	2	2
NA	0	1	0	2	0	3	0	2
MLRv2	4	0	2	0	11	0	4	0
Total messages	9	4	8	8	16	8	8	6
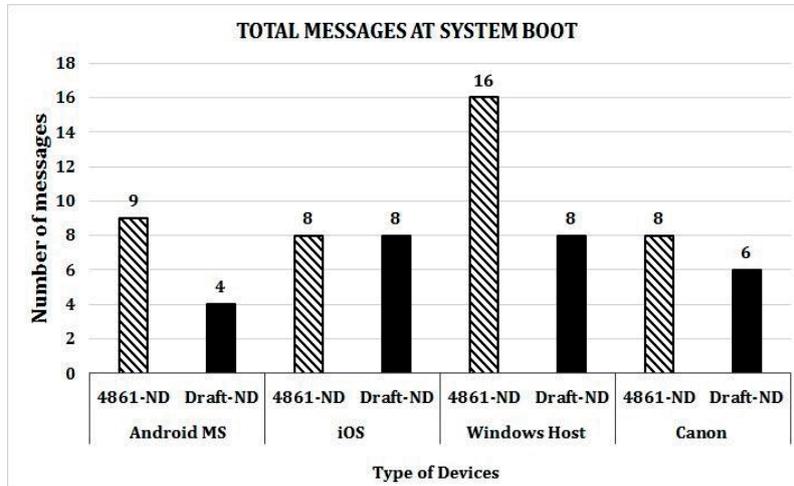
Fig. 47. Total Multicast messages exchanged during connection set up for Android MS, iOS device, Windows host and Canon device irrespective of unicast or multicast
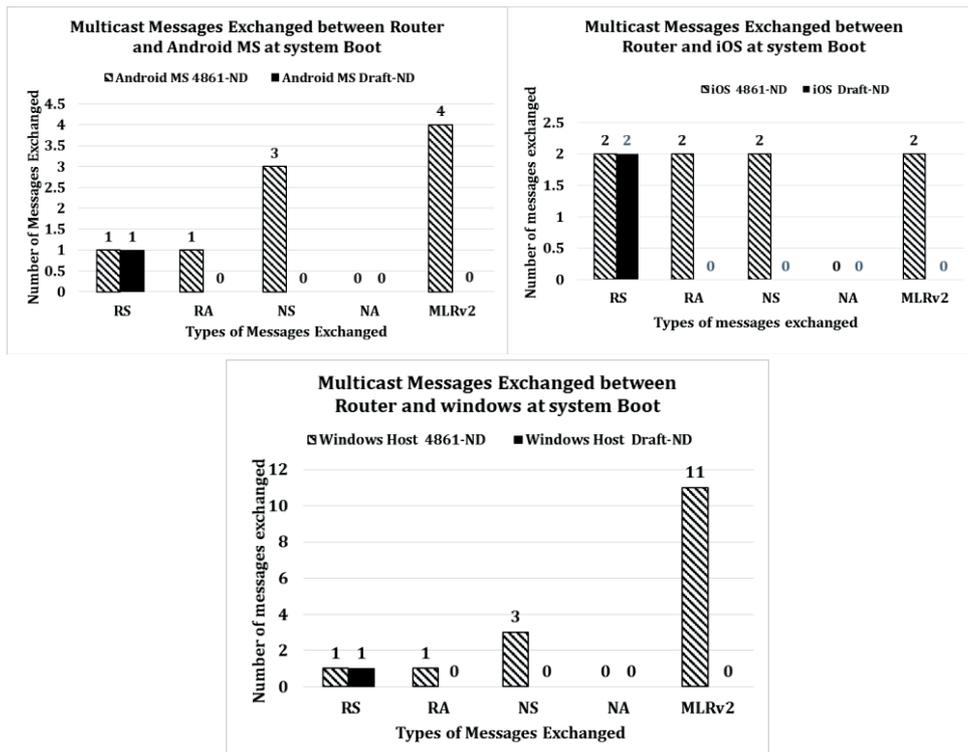
Fig. 48. Multicast messages exchanged during connection set up for Android MS, iOS device and Windows host

Now considering the total multicast messages exchanged during system boot up for different devices as in legacy ND and efficient ND is shown in Fig. 48. The results closely follow the behavior mentioned in section 6.2.1 of chapter 6 for the legacy ND and the expected results for modified ND shows a considerable improvement in comparison with these results.

The experiment is also performed for a Wi-Fi printer device to check the ND behavior of an M2M device. This printer is not a battery operated device but powered directly. The device could not be analyzed for the multicast messages as the implementation will not produce a substantial effect in its battery life.

Fig. 49. Total multicast messages exchanged during the connection setup for Canon Wi-Fi Printer

The Fig.49 shows the ND behavior of the printer during connection set up. It is to be noted that unlike other wireless devices discussed until now, the printer does not bear a privacy address. Instead it holds a EUI-64 based link local address (*fe80::1a0c:acff:fe13:f44c*) derived from its hardware address (*18:0c:ac:13:f4:4c*) and a EUI-64 based IPv6 global address (*2001:dbb:0:1:1a0c:acff:fe13:f44c)* with the advertised prefix. It sends 4 MLDv2 messages to the multicast listener group. This behavior can be expected to be the general nature of any M2M device.

7.1.2 Scalability of the network

Assuming that when the Windows host enters the network it makes the connection and for the next 20 minutes no other hosts enters the network. During this duration the advertised default router lifetime is chosen to be the maximum allowed value (65535 seconds) to reduce the periodic multicast RAs. The total number of RAs exchanged is 4 when only Windows host join the network as shown in Fig. 50 and the total number of multicast messages exchanged in the network during this duration is 23 as

71

shown in Fig. 51. With the modified ND, the number of RAs multicast is zero and the total multicast messages exchanged is 1. The total multicast messages includes the MLDv2 which can be varied for different implementations.

Further when MS, iOS device and printer are connected to the network one after the other, the number of periodic RAs does not show a rapid increase because the router lifetime is maximum in the legacy ND. However the multicast messages exchanged within the network to set up the connections for the newly added devices increases instantly. Meanwhile, the efficient ND shows that the number of RAs will remain zero and the total multicast messages increases with the number of nodes. The value can increase by 1 sometimes or by 2 at some other times depending on the device specifications as shown in Fig. 51.
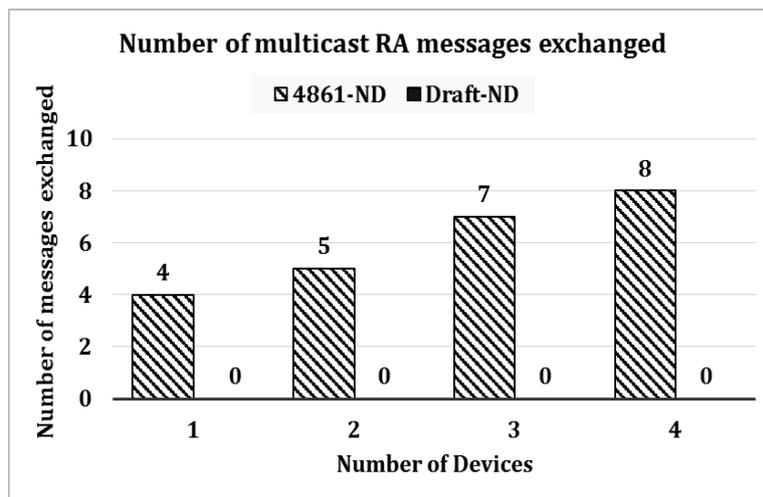
Fig. 50. Total multicast RA messages exchanged during scalability check of the network

The comparison for RA messages is shown in Fig. 50, whereas the comparison for multicast messages is shown in Fig. 51.
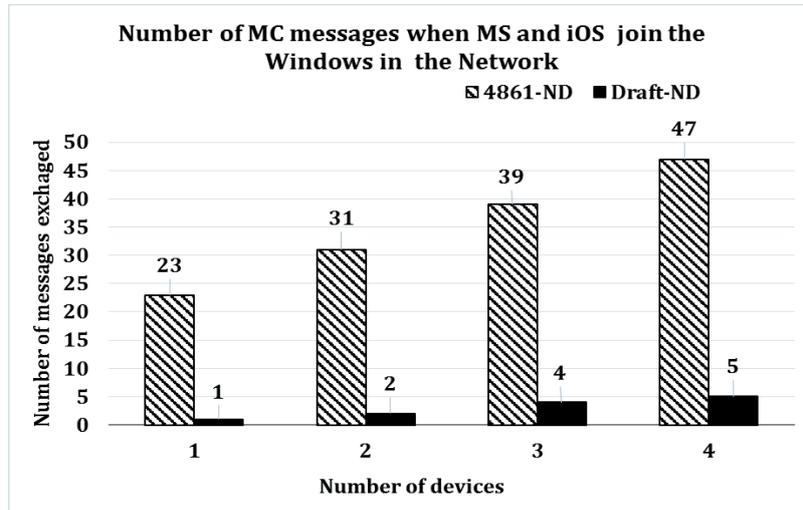
Fig. 51. Total multicast messages exchanged during scalability check of the network

It is essential to mention that these multicast messages will not be intended for the clients but instead for the routers on the attached link. Therefore we can draw a conclusion that the battery operated client devices can save their power with the modified efficient ND. The routers on the link are the nodes that are involved in the multicast traffic and they are always left plugged in to the power supply. Thus draft can significantly be beneficial in terms of scalability of the network.

7.1.3 Transient Behavior of the network

In this scenario, implementation is assumed to have Windows host already joined to the network as mentioned in 7.1.2. After 20 minutes of duration, the nodes starts joining the network one after the other in the order MS first, iOS next and finally canon printer. The implementation set up choses an *AdvDefaultLifetime* of *65535 seconds* to reduce the periodic RAs. After a few seconds the network shows momentary behavior due to the already joined nodes re-joining and leaving the network. The Fig. 52 shows the number of multicast messages exchanged by the nodes at different circumstances.
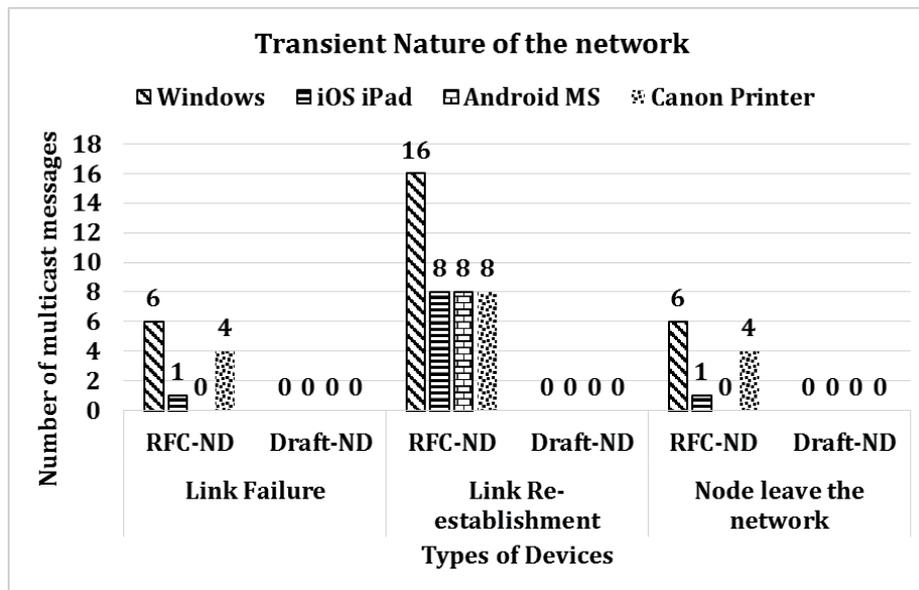
Fig. 52. Multicast messages exchanged during transient nature of the network

Now considering a scenario where the following events occurred within 10 minutes duration in the network which includes all the nodes:

- MS leaves the network
- iOS experience a failure of link
- iOS re-joins
- Windows leaves the network.
- Printer leaves the network
- Printer re-joins

In such a scenario, it is desirable to compare the number of multicast messages transmitted, as these messages would reach all the devices that already joined the network. As seen in Fig. 52, if the MS leaves the network no messages are transmitted. Later when iOS lose its link connection it sends 1MLDv2 message and when it rejoins it sends 8 multicast messages. After a while, when Windows host leaves the network it multicasts 5 or 6 MLDv2 messages followed by the canon printer leaving the network multicasting 4 MLDv2 messages. Finally when the printer re-establishes the connection, it sends 8 multicast messages which makes a total of 27 messages being multicast within the network.

If the same scenario is analyzed for the efficient ND, the devices joined will be registered to the default router for the *AdvDefaultLifetime* which is *65535 seconds*. During this period the nodes need not send any messages even if its link fails, or it re-joins the network. When the nodes leave the network it sends one NS with ARO option that indicates the router registration lifetime to zero is send as unicast message. Hence neither of the messages would be multicast messages. Hence zero multicast messages are exchanged. Thus the address registration procedure in the efficient draft is a great advantage to avoid unnecessary message transfers.

7.1.4 Multicast messages exchanged in the network

Based on the multicast messages exchanged in the network during the experiments conducted, certain estimates are shown below. The number of multicast RA messages exchanged is analyzed for a duration of 5, 10, 15, 20 minutes by varying the *Min/MaxRtrInterval*. The *AdvDefaultLifetime* is also varied to the maximum allowed value to verify the changes in the periodic exchange of RA updates. A comparative study for these values with that in case of efficient ND is also analyzed theoretically.
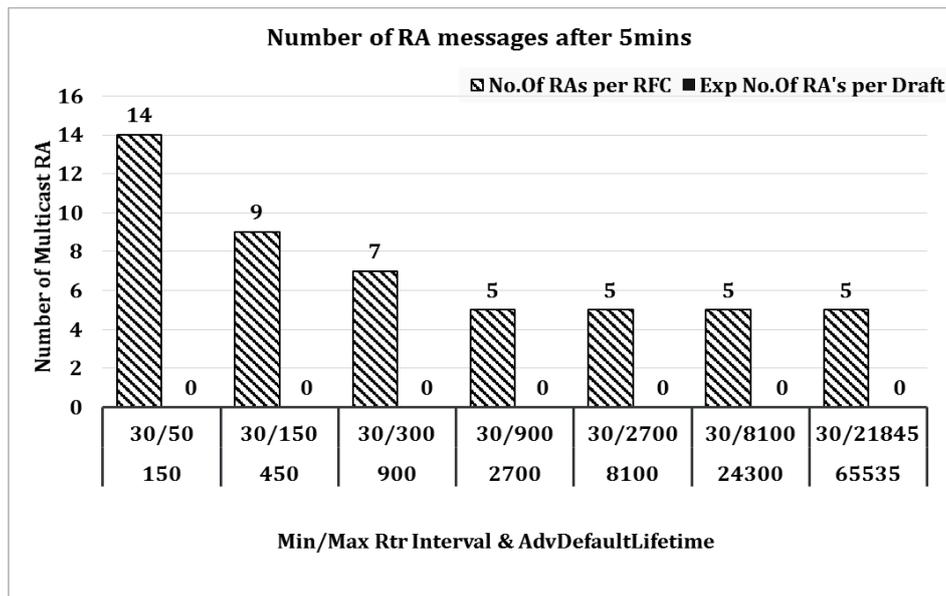
Fig. 53. Number of RA messages in the network after 5 minutes
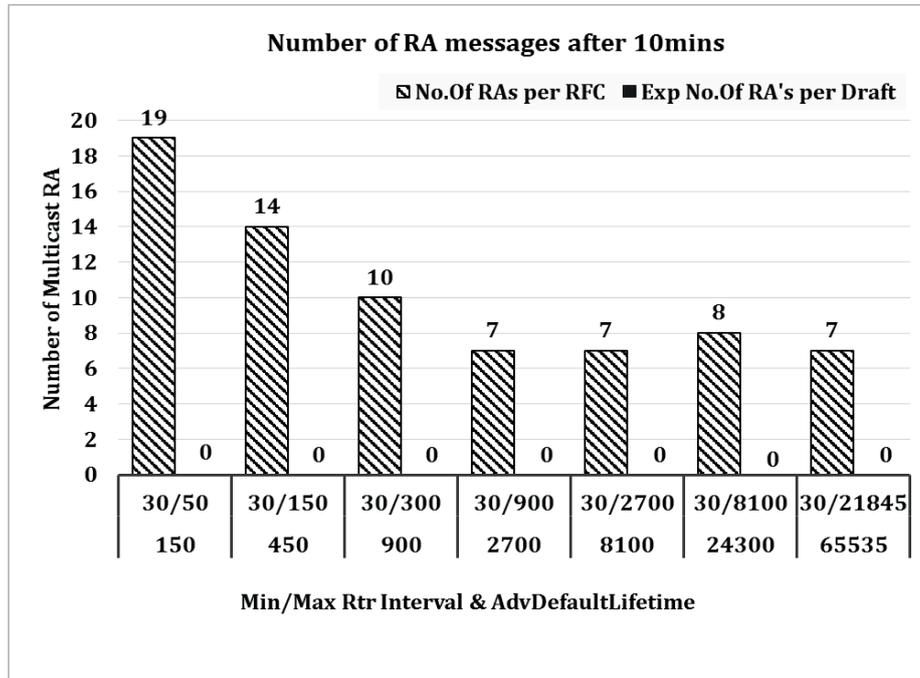
Fig. 54. Number of RA messages in the network after 10 minutes
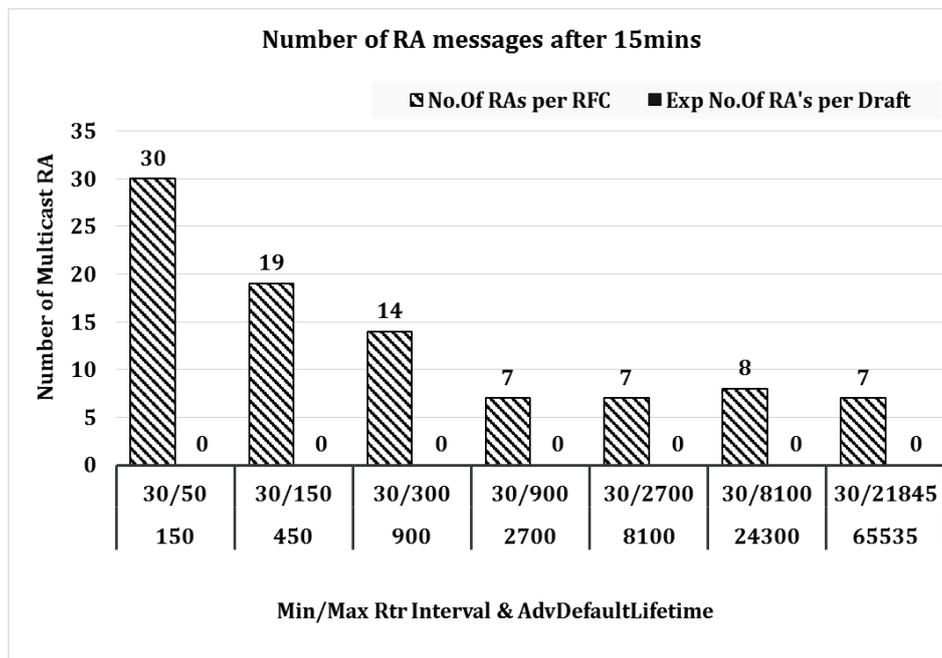
Fig. 55. Number of RA messages in the network after 15 minutes
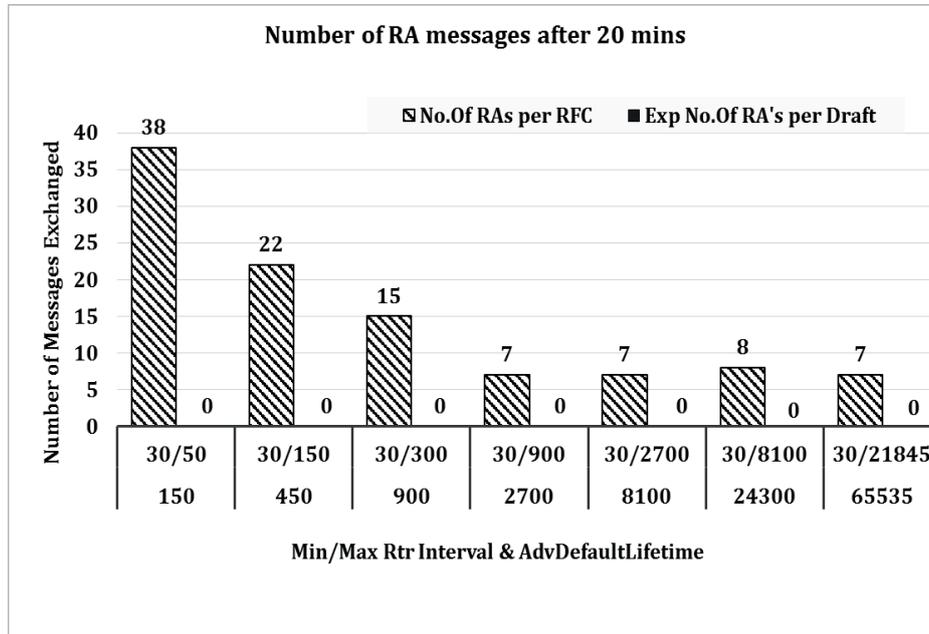
Fig. 56. Number of RA messages in the network after 20 minutes

From the observations in Fig. 53, Fig. 54, Fig. 55 and Fig. 56, we can see that in the case of the legacy NDP, increasing the difference between the *MinRtrAdvInterval* and the *MaxRtrAdvInterval* will decrease the number of periodic RA updates in the network. That means increasing *AdvDefaultLifetime* reduces the number of RA updates in the network. This can reduce the number of unnecessary signals in the network. Hence for an *AdvDefaultLifetime* of *65535 seconds* and *MaxRtrAdvInterval* of *21845 seconds,* the minimum number of RAs exchanged from the router to a node will be given by the *k factor* mentioned in the chapter 6 section 6.3. Thus k for this case is calculated as shown in (2).

$K = AdvDefaultLifetime/MaxRtrAdvInterval$ (2)

$= 65535/21845 = 3$

There will be a minimum of 3 periodic RA updates in the network. These many updates are intended to be sufficient to keep the nodes updated about the changes in the network.

The authors of this thesis would like to assume the M2M use case scenario used in [3]. The author of [3] considers a typical M2M network situation that contains:

- A single Packet Data Network Gateway(PGW) node
- 10,000,000 simultaneous IPv6 M2M devices for PGW
- Infrequent communication from network to the user device
- If state is IDLE, when a packet reaches user equipment, the paging messages will be triggered to set the device to CONNECTED state.
- 100 Base stations with 10 Mobility Management Entity(MME) Tracking Areas each making 10*100 base stations=1000 base stations for a user device
- Maximum allowed values for the router advertise intervals (*MinRtrAdvInterval* and *MaxRtrAdvInterval*). In this thesis work, the value of *MaxRtrAdvInterval* is assumed as *21845 seconds* due to the benefits of reducing the periodic RAs to the maximum possible number [16].

Based on these assumptions, a calculation is made for the RA updates in the network based on the connectivity state changes and the resource allocation.

The analysis is shown in (3).

$$MaxRtrAdvInterval \; = 21845 \; seconds \qquad\qquad (3)$$

$$MinRtrAdvInterval \; = 0.75*21845 = 16383.75 \; seconds$$

$$Average \; RA \; interval = (MaxRtrAdvInterval + MinRtrAdvInterval)/2$$

$$=(21845+16383.75)/2=19114.375 \; seconds$$

$$RAs \; /second \qquad\quad = No.of \; Devices/ \; Average \; RA \; interval$$

$$= 10,000,000/ \; 19114.375$$

$$= 523.166$$

Thus if periodic updates are allowed in the network, for every 1 second approximately *523* RAs will be exchanged. So for a 1000 base stations, around *523000* paging messages would be initiated in the network per second. Since the communication is infrequent, an update like this would trigger the states of all the nodes in the network including Serving Gateway (SGW), MME, base station, and user device. Therefore, approximately 4000(i.e., 523*2*4) state changes happen per second within the network. This illustrates that even if we reduce the number of multicast RAs as much as possible, the effect of periodic updates on network resources as well as traffic is very high[3].

A better solution is obtained while analyzing the efficient ND behavior. According to this modified protocol, multicast RA does not exist in the network as shown Fig. 53, Fig. 54, Fig. 55 and Fig. 56. This can save both bandwidth and energy to use them for the actual communications.

Even though we completely avoid the periodic multicast RAs, still the other multicast signaling that is happening in the network during DAD and other processes should be analyzed. The Fig. 57, Fig. 58, Fig. 59 and Fig. 60 are the observations for the total multicast messages in the network for the durations 5,10,15,20 minutes. The nodes Android MS, iOS device and the Windows host joins the network randomly.
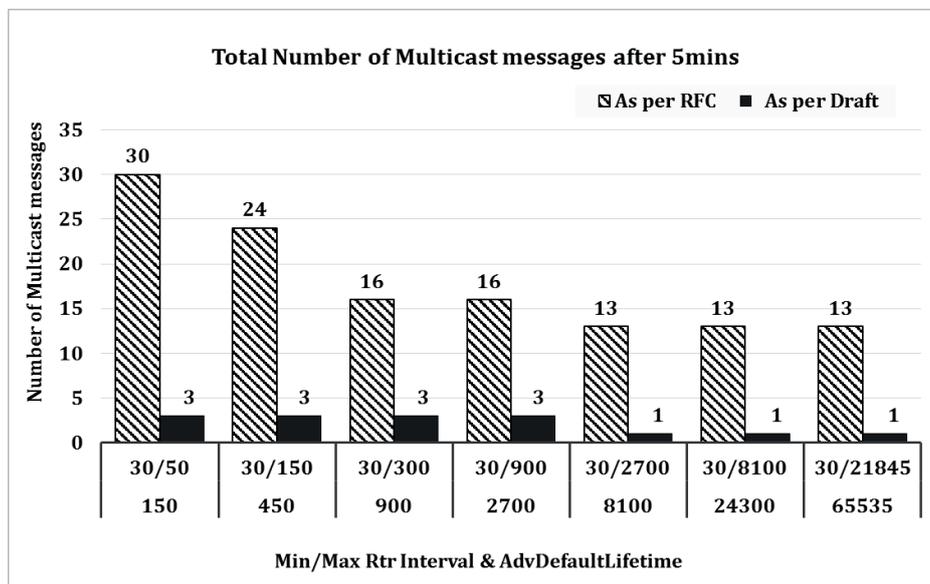
Fig. 57. Total number of multicast messages in the network after 5 minutes
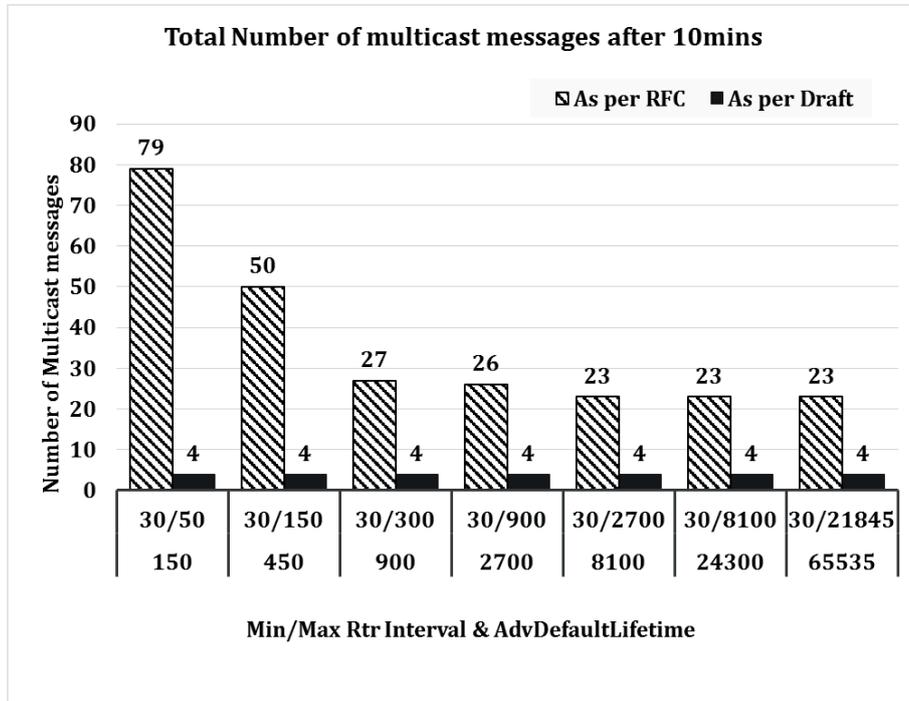
Fig. 58. Total number of multicast messages in the network after 10 minutes
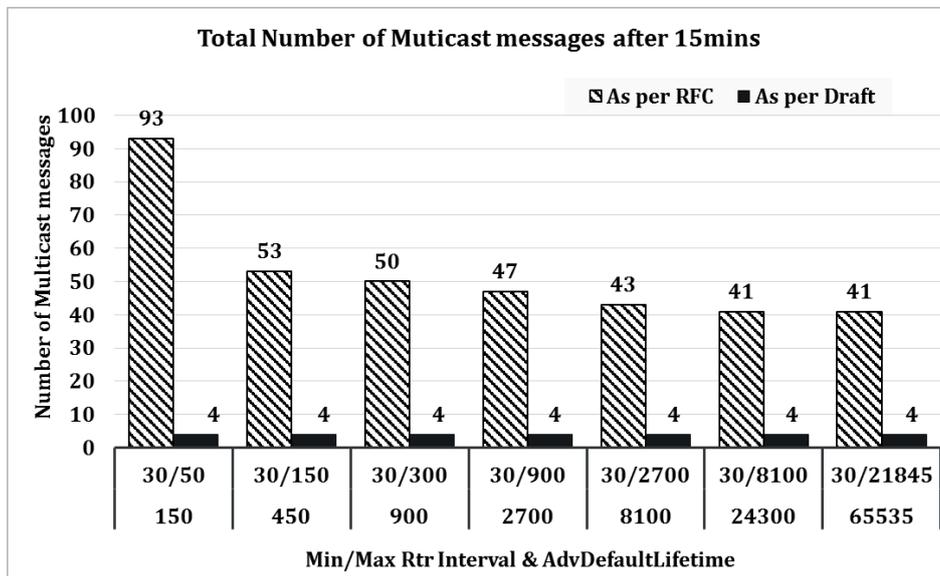
Fig. 59. Total number of multicast messages in the network after 15 minutes
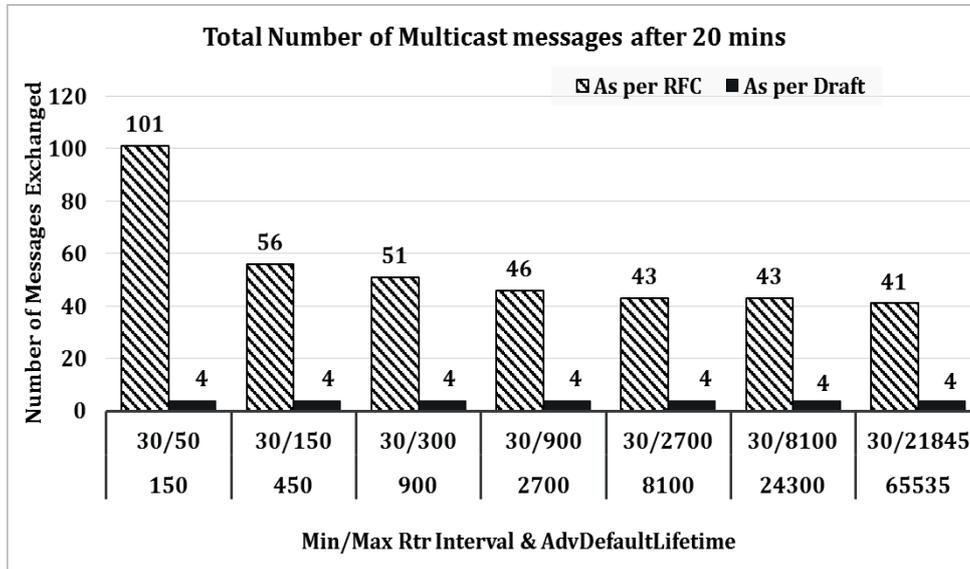
Fig. 60. Total number of multicast messages in the network after 20 minutes

The experimental results shown in Fig. 57, Fig. 58, Fig. 59 and Fig. 60 shows that multicast messages reduces when the frequency of messages decreases in the network. Comparing the results of legacy NDP with the proposed efficient ND, the draft results shows a better theoretical performance. The figures Fig. 61, Fig. 62 and Fig. 63 shows the number of multicast RA messages for the wireless devices individually whereas Fig. 64, Fig. 65 and Fig. 66 gives an idea about the number of multicast messages for each of the wireless device. Although in the real network situations these numbers highly differ, these results serves the purpose of giving the reader an understanding of the intensity of multicast messages in the network.
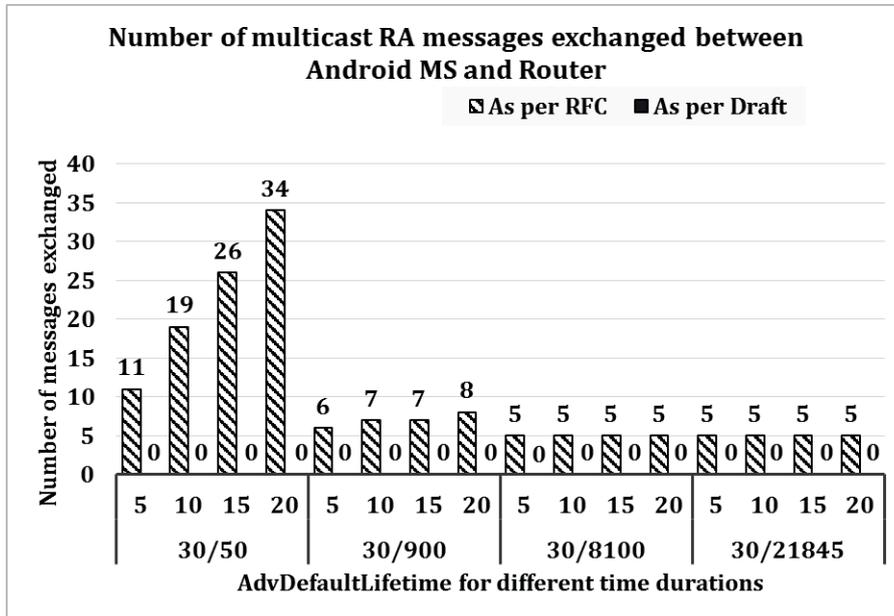
Fig. 61. Number of multicast RA messages exchanged between Android MS and router
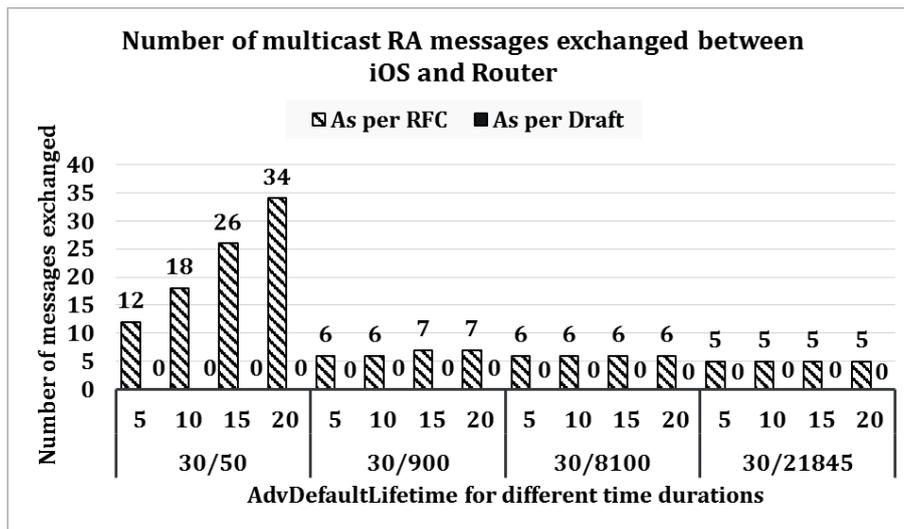
Fig. 62. Number of multicast RA messages exchanged between iOS device and router

82
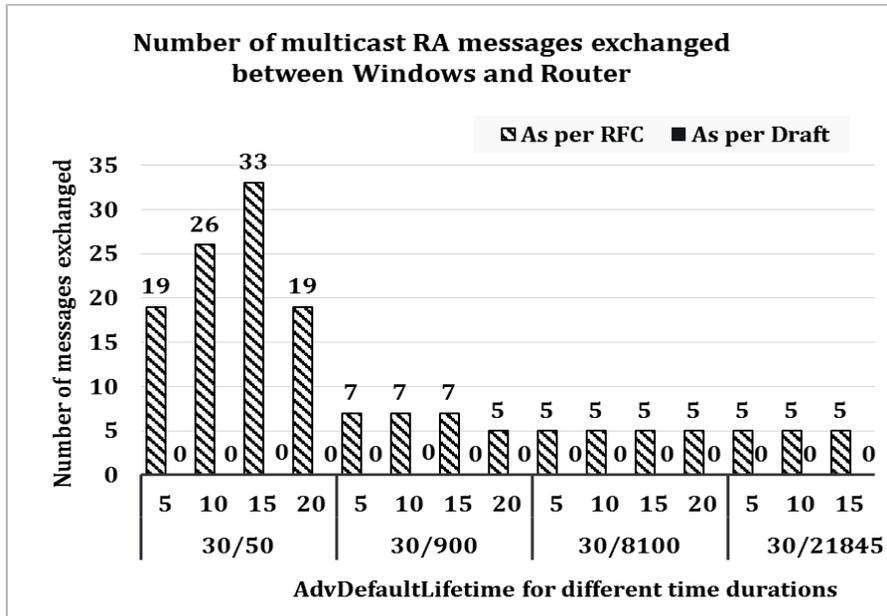
Fig. 63. Number of multicast RA messages exchanged between Windows host and router
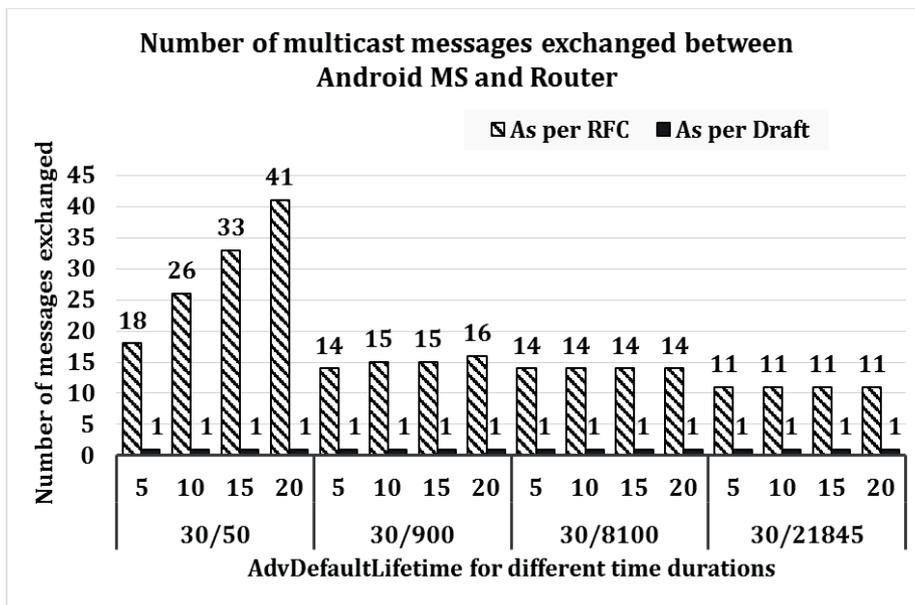
Fig. 64. Total Number of multicast messages exchanged between Android MS and router
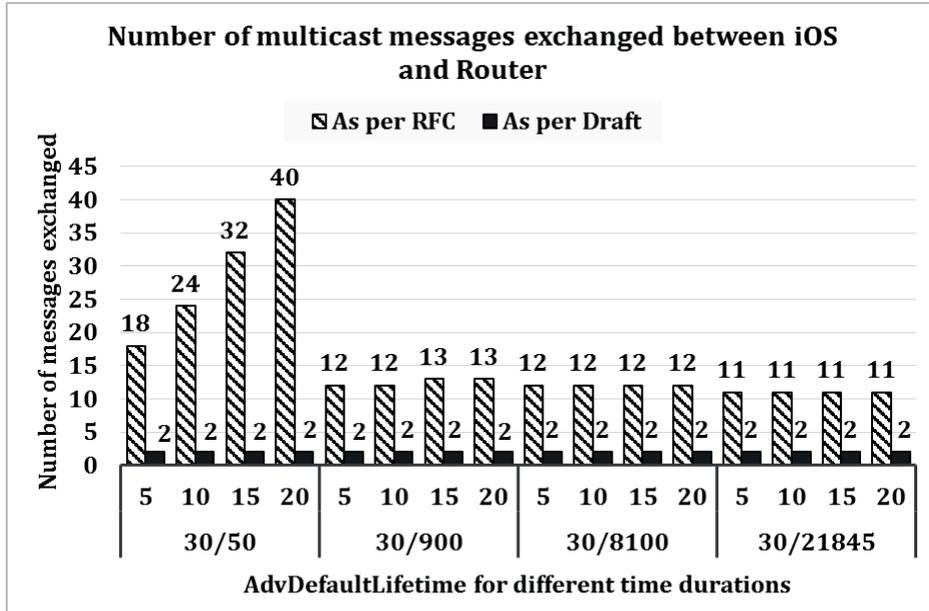
Fig. 65. Total Number of multicast messages exchanged between iOS device and router
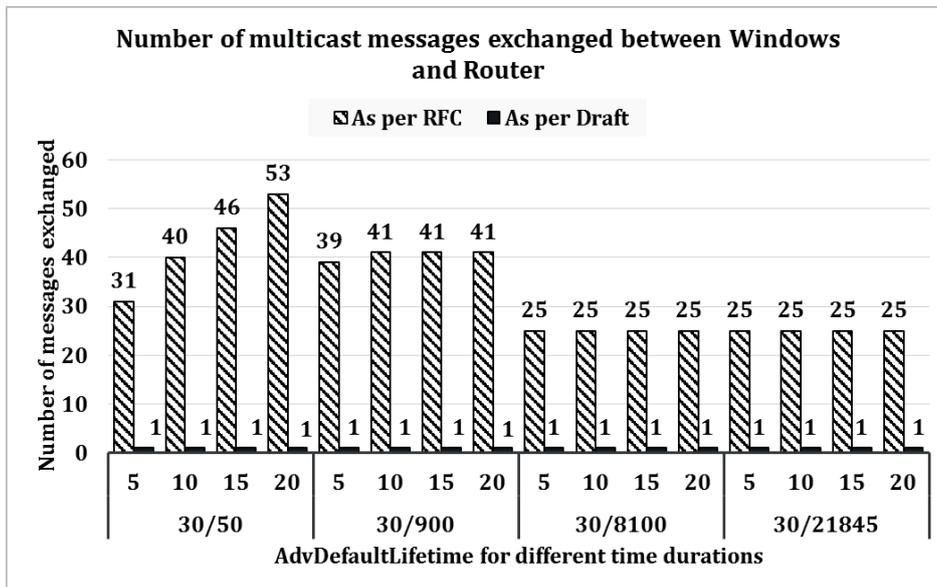
Fig. 66. Total Number of multicast messages exchanged between Windows host and router

7.1.5 Sleep and wake up scenario

As mentioned in chapter 6 section 6.2.5, the sleep and wakeup test case is not performed but a theoretical analysis of efficient ND for this test case is conducted based on [15].

According to [15], when a node goes to sleep, any attempt to assign its IPv6 global address to another node joining the network succeeds if the node has a power saving mechanism that keeps the radio of the device switched OFF. However if the node has a power saving technique that listens to the DAD messages and responds to it, the address assignment fails.

When the node wakes up, the node will not perform any DAD check to find the uniqueness of its address either because it listened to the DAD messages during sleep or because the legacy NDP don't handle this case. Now the case is that if the node listened to the DAD check the communication goes on. On the other hand, the IPv6 address of the node would have been allotted to another device and all the communications addressed for this previous node will be directed to the node that owns the addresses now. The situation changes once the router entry is flushed. This can be due to a manual operation or due to bootstrapping of the router. After any of these process, the router will ask for details from the nodes on the network and the node will get back its address as its entry will be the latest in its NCE.

This case is compared with the scenario that uses an efficient ND behavior. The nodes make registration with its default router and goes to sleep. When node wakes up it sent a NS+ARO message to refresh its registration lifetime. The node sent a unicast message which does not disturb the other nodes in the network and thus this procedure avoids lots of complications and multicast messages in the network.

7.2 Summary

From the experimental results, it is seen that even though the experiments conducted in this work is far from the ideal network conditions, the theoretical assumptions made in chapter 6 are satisfactorily followed. A summarized result analysis of the multicast RA messages and the total multicast messages exchanged is shown in Fig. 67 and Fig. 68.
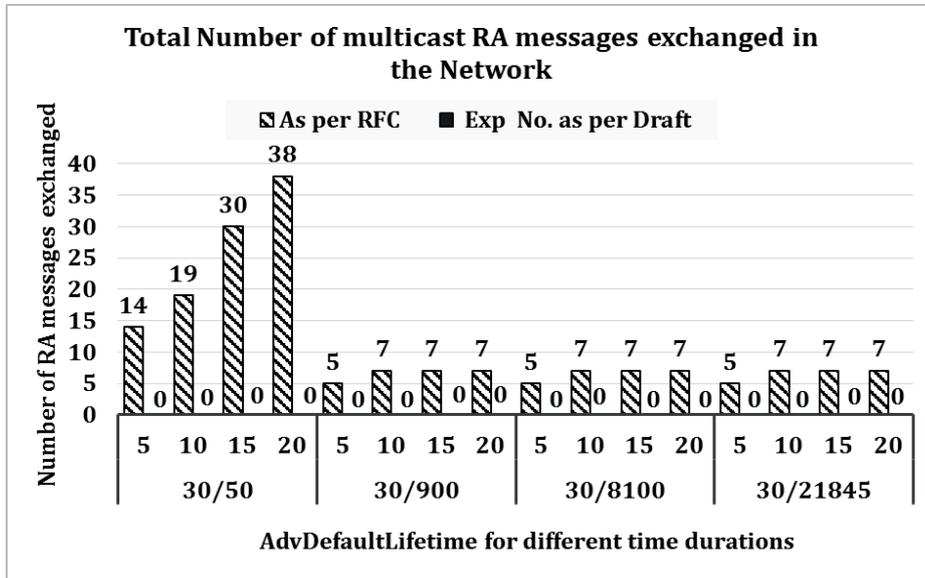
Total Number of multicast RA messages exchanged in the Network

As per RFC   Exp No. as per Draft

Fig. 67. Comparison of total RA messages in the network as per RFC 4861-ND and expected number of RA messages as per Draft-ND for different time durations

Total Multicast Messages in the Network

As per RFC   As per Draft
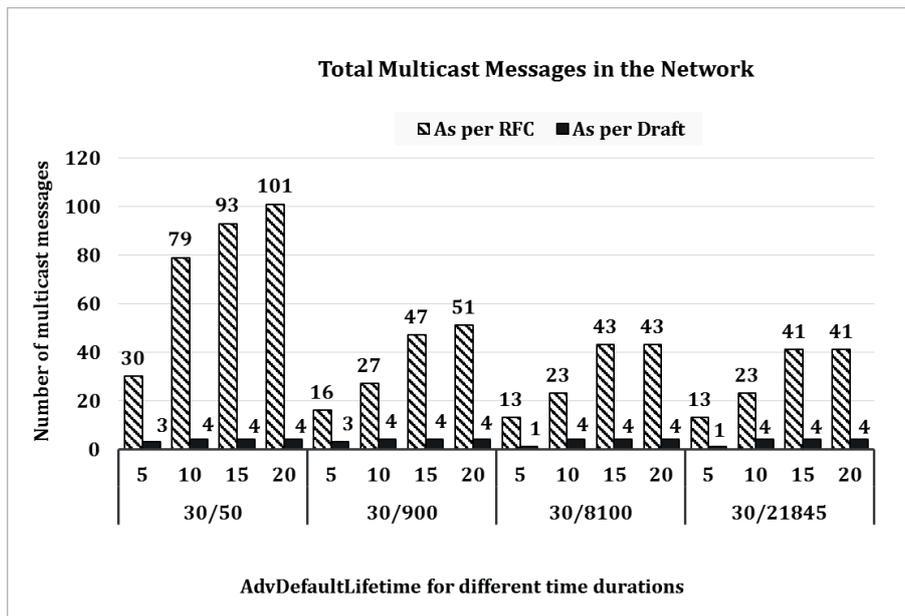
Fig. 68. Comparison of total multicast messages in the network as per RFC 4861-ND and expected number of messages as per Draft-ND for different time durations

The comparison of these results with the expected results of efficient ND draft theoretically shows that, the draft ND is beneficial for the wireless networks to a very great extent. Thus the basic comparison of legacy NDP and the modified efficient NDP can be as shown in Fig. 69 and the complete operations can be shown as a flow chart as shown in Fig. 70.
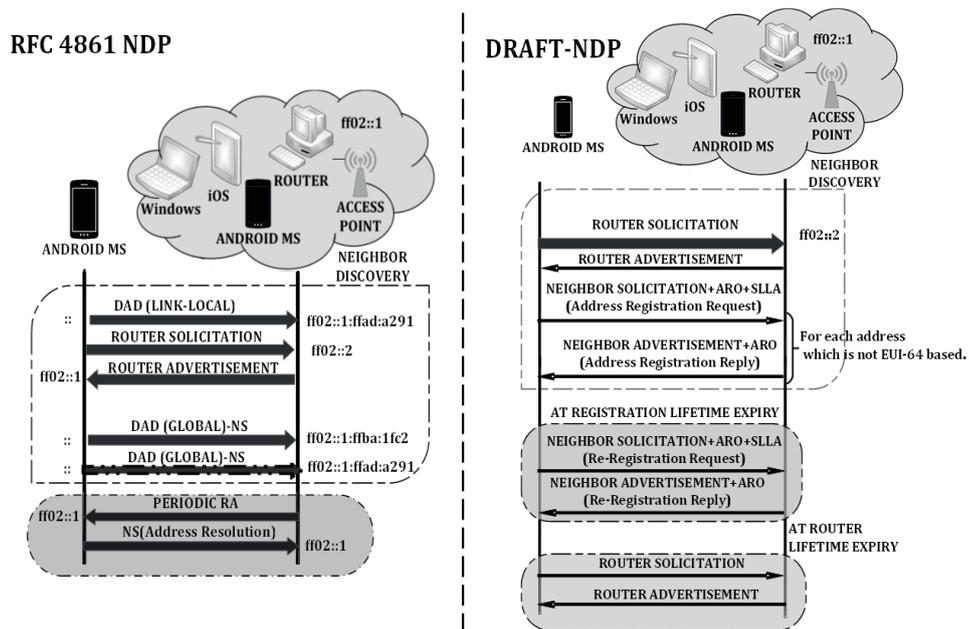
Fig. 69. Basic theoretical idea of the comparison between the legacy NDP and the efficient-NDP

IDS: Identifier Name space
UIID: Unique Interface Identifier
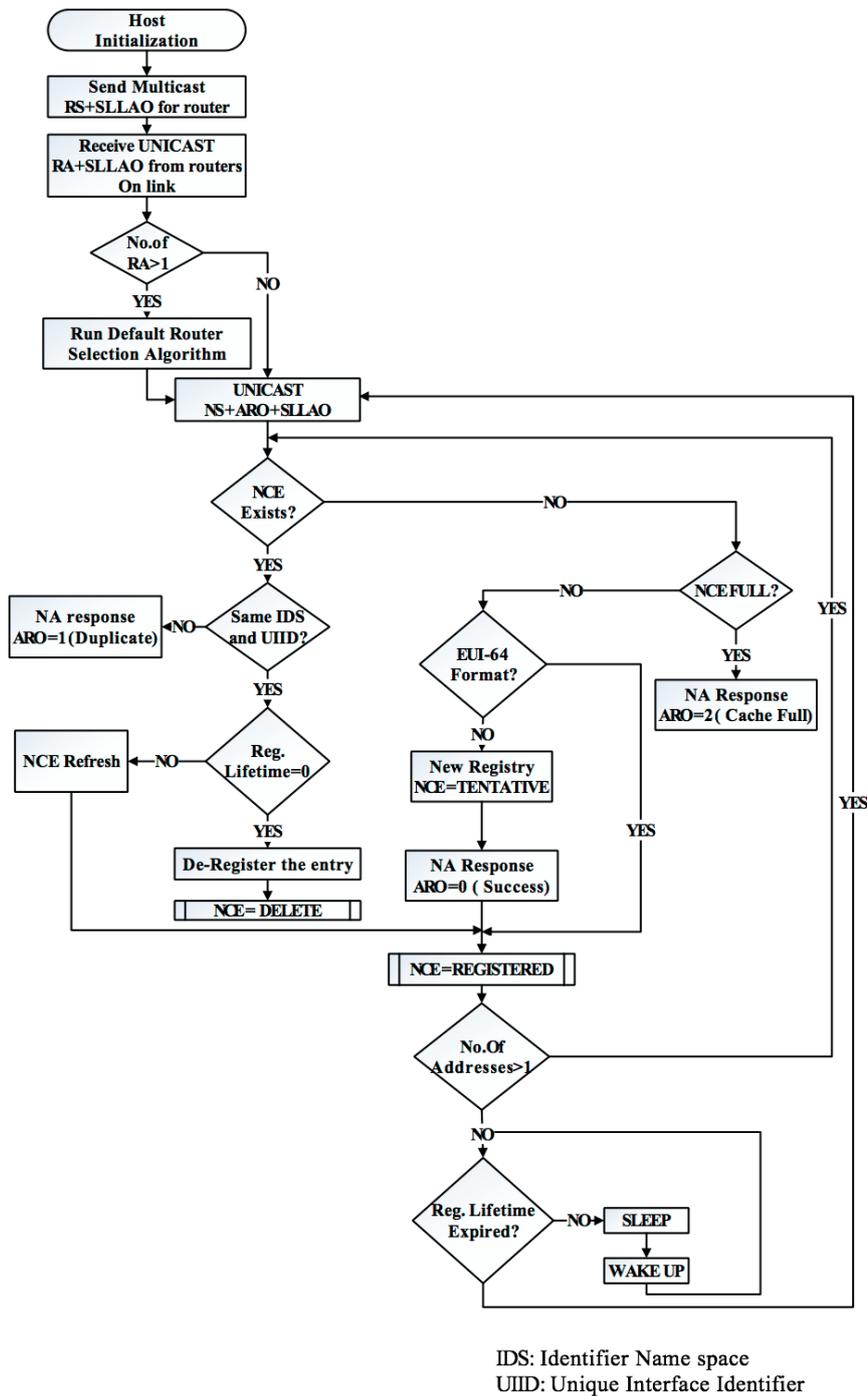
Fig. 70. Flowchart of the basic ND operation as stated in the efficient-ND draft

88

CHAPTER 8

8 Conclusions

The efficient-ND draft principally aims at making battery operated mobile hosts more energy efficient by letting them go undisturbed from unwanted control messages during their inactive 'sleep modes'. The results of the implementation carried out for the legacy NDP in chapter 7 while compared with the theoretical analysis of the modified protocol is shown to have a vast number of unnecessary multicast messages both for the scalability and transient behaviors of the network. Theoretically, a reasonable number of these messages are avoided comparatively in efficient ND whereby the battery life of the devices is anticipated to be saved. This is a highly promising advantage for the vendors as well as the customers.

8.1 Summary

Following summary of the theoretical analysis of the draft for the different use cases in comparison to the empirical results of legacy NDP will give a complete insight of this work.

8.1.1 IPv6 ND Behavior (TC1)

TABLE 5. PERCENTAGE OF MESSAGES SAVED AT THE CONNECTION SET UP FOR DIFFERENT SYSTEMS.

Devices	% of messages saved (approx.)
Android	88%
iOS	75%
Windows	93%
Canon printer	87%

Theoretically the draft is shown to give a better performance compared to the existing NDP during the connection set up as supported by the Table 5. Thus the registration process in the draft saves messages during the IPv6 ND behavior and the DAD procedure.

8.1.2 Scalability of the network (TC2)

The order of the devices entering the network is given in detail in section 7.1.2.

TABLE 6. PERCENTAGE OF MESSAGES SAVED DURING THE SCALABILITY OF THE NETWORK

Number of Devices	% saved for multicast RA	% saved for total multicast
1 (Windows enters)	100%	95%
2 (MS enters)	100%	93%
3 (iOS enters)	100%	89%
4 (Canon enters)	100%	89%

8.1.3 Transient Behavior of the network (TC3)

TABLE 7. PERCENTAGE OF MESSAGES SAVED DURING TRANSIENT NATURE OF THE NETWORK

Device	Link Failure	Link Re-establishment	Node leave the network
Windows	100%	100%	100%
iOS iPad	100%	100%	100%
Android MS	0%	100%	0%
Canon Printer	100%	100%	100%

The first and foremost goal of this thesis work as seen in chapter 1 section 1.3 was to investigate if the draft is beneficial for the scalability and transient nature of the network. The theoretical verification as shown in Table 6 and Table 7 establishes that the draft is highly beneficial to make a

remarkable effect on these nature of the network by saving messages within the network.

8.1.4 Multicast messages exchanged in the network (TC4)

The evaluation of the draft as seen from Table 8 shows that it reduces the multicast traffic by cutting down the flow of unwanted signals in the network, although the RS and the emergency update RAs still remain unavoidably multicast. This will definitely increase network efficiency which is inversely proportional to the number of control messages flowing in the network for each data packet.

TABLE 8. PERCENTAGE OF MULTICAST MESSAGES SAVED IN THE NETWORK

Min/Max RtrInterval	Time duration (minutes)	% of multicast RA messages saved	% of total multicast messages saved
30/50	5	100%	90%
	10	100%	95%
	15	100%	96%
	20	100%	96%
30/900	5	100%	81%
	10	100%	85%
	15	100%	91%
	20	100%	92%
30/8100	5	100%	92%
	10	100%	83%
	15	100%	91%
	20	100%	91%
30/21845	5	100%	92%

10	100%	83%
15	100%	90%
20	100%	90%

8.1.5 Sleep and wake up scenario (TC5)

As mentioned in the section 7.1.5 the node does not lose its address during its sleep cycle as it is registered with the default registrar. It updates itself while in the active mode by sending unicast NS to the registrar. This means that the draft saves the multicast messages sent to save the node's addresses during its idle mode and also helps to save the battery by allowing to complete their sleep cycle. Thus the draft is expected to show 100% efficiency in handling this situation.

For a host in M2M and IoT networks (hosts that send short burst of data once in a while and stay silent for a longer period of time), the unsolicited periodic RA packets unnecessarily consumes processing power for paging and causes network traffic. In such a case, to reduce energy usage and network resources, the hosts will have to release the resources allocated to them and also avoid responding to the unsolicited messages while being in sleep mode. Thus the analysis with respect to the M2M scenario shown in section 7.1.4 provides the insight that reducing the periodic RAs is not the complete solution to attain an efficient battery saving and resource saving. Instead these RA updates should be totally avoided and using the Address Registration Option (ARO) in the efficient-ND draft can help the node to complete the sleep cycle to save messages and energy.

8.2 Conclusion

When observing the theoretical working of the modified protocol with the legacy protocol (mainly with respect to IoT and M2M networks), the modified protocol focuses on more centralized control at the router. The hosts are given the freedom to choose their default router, but from then the router takes charge and sends all control messages on the host's behalf. But centralization, can also lead to bigger problems in case of device or link failures, which is almost always the case with wireless networks. The draft does provide an EPOCH mechanism for data recovery in case of router failure and overrides this disadvantage. Hence the draft proposes a good

balance between centralized networks and decentralized networks and can efficiently exchange the control messages.

The main findings of this thesis work while analyzing the efficient-ND draft are:

- TC1 shows at least 85% message saving for the wireless devices
- TC2 shows approximately 91% message saving for the total multicast messages and 100% message savings for multicast RA messages for the wireless devices
- TC3 shows approximately 100% savings of messages for the link failure, link re-establishment and while a node is leaving the network
- TC4 shows 100% message savings for multicast RA messages and 84% savings for total multicast messages
- TC5 shows 100% efficiency in saving the multicast messages

9 Future Work

This thesis work concludes with the theoretical analysis of the efficient ND protocol. Due to the broadness of the work, several aspects have been left out of the scope of this thesis work. The test case scenarios for the efficient ND analyzed in this work is purely theoretical. In particular, the sleep and wake up scenario is not compared due to the fact that legacy NDP does not favor this scenario. The implementation part is not performed completely and so this scenario cannot be interpreted. Hence a natural logical step would be to complete the implementation. The complete implementation would require recreating a new binary executable for the RADVD with the changes and creating a new subroutine for initiating and maintaining the registrations. The host kernel has to be reprogrammed to decode the packets accordingly and respond to the router's requests.

Finally creating a wireless network with unsteady links and seeing how many control messages flow with respect to each host would be the next future task. Another task would be to see how the network gets updated with control messages when several hosts temporarily join the network and leave the network. In such a case, the implementation results can be compared to the implementation results of the legacy NDP and a conclusion can be drawn.

 In order to measure the percentage of battery saved, the implementations should be performed for extended durations or may be for 24 hours with the dedicated devices. The performance tests must be conducted for battery operated M2M devices to check the feasibility of the efficient ND protocol on such devices.

References

[1]T. Narten, E. Nordmark, W. Simpson and H. Soliman, "*Neighbor Discovery for IP version 6 (IPv6)*"RFC 4861, September 2007.

[2]S. Chakrabarti, E.Nordmark, P. Thubert, M. Wasserman, "*IPv6 Neighbor Discovery Optimizations for Wired and Wireless Networks*", [work in progress], August 2015.

[3]F. Garneij, S. Chakrabarti, S.Krishnan, "*Impact of IPv6 Neighbor Discovery on Cellular M2M Networks*", [work in progress], July 2014.

[4]E. Vyncke Ed., P. Thubert, E. Levy- Abengnoli, A. Yourtchenko, "*Why Network-Layer Multicast is Not Always Efficient At Data link Layer*", [work in progress],February 2014.

[5]GitHub guide for Open Source software's.
https://help.github.com/articles/set-up-git/

[6]Linux Ipv6 Router Advertisement Daemon
http://www.litech.org/radvd/

[7]Z. Shelby Ed., S. Chakrabarti, E.Nordmark, C.Bormann, "*Neighbor Discovery Optimizations for IPv6 over Low-Power Wireless Personal Area Networks (6LowPANs)*", RFC 6775, November 2012.

[8]S. Thomson, T. Narten, and T. Jinmei, "*IPv6 Stateless Address Auto configuration*" RFC 4862, September 2001.

[9]S. Deering, W. Fenner, B. Haberman, *"Multicast Listener Discovery (MLD) for IPv6*", RFC 2710, June 2004.

[10]R. Vida Ed. and L. Costa Ed., "*Multicast Listener Discovery Version 2 (MLDv2)*", RFC 3810, June 2004.

[11]F. Shearer, "Power Management in Mobile Devices", ISBN 978-0-7506-7958-9.

[12] R. Zheng, J.C. Hou and N. Li, "Power management and power control in wireless networks", Ad Hoc and Sensor Networks, Nova Science Publishers, 2004.

[13] C. Betancourt, K. Shin, " *A white paper on Power-saving techniques lead to ultra-low-power processors for battery-operated devices*", April 2013.

[14] C.Bormann, M.Ersue and A. Keranen, "*Terminology for Constrained-Node Networks*", RFC 7228(Informational), May 2014.

[15] A. Sood, "Test *result analysis of IPv6 Neighbor Discovery in a simple Wireless network*", [work in progress], March 2015.

[16] S. Krishnan, J. Korhonen, S. Chakrabarti, E.Nordmark, A. Yourtchenko, "Support *for adjustable maximum router lifetimes per-link*", [work in progress], July 2015.

[17] T. Narten, E. Nordmark, W. Simpson, "*Neighbor Discovery for IP Version 6 (IPv6)*", December 1998.

List of Figures

100

List of Tables

List of Acronyms

IoT	Internet of Things
M2M	Machine to Machine communication
IPv6	Internet Protocol version 6
ND	Neighbor Discovery
NDP	Neighbor Discovery Protocol
ICMP	Internet Control Message Protocol
ICMPv6	Internet Control Management Protocol version 6
SLAAC	Stateless Address Auto-Configuration
DHCPv6	Dynamic Host Configuration Protocol
3GPP	Third Generation Partnership Project
RADVD	Router Advertisement Daemon
RA	Router Advertisement
RS	Router Solicitations
NA	Neighbor Advertisement
NS	Neighbor Solicitation
MTU	Maximum Transmission Unit
DAD	Duplicate Address Detection
EAH	Efficiency Aware Host
NEAR	IPv6 ND-efficiency-aware Router
ARO	Address Registration Option
NCE	Neighbor Cache Entry
UIID	Unique Interface Identifier
TID	Transaction Id
EUI	Extended Unique Identifier
IDS	Identifier name Space
SLLA	Source Link Layer Address
NUD	Neighbor Unreachability Detection
MLDv2	Multicast Listener Discovery version 2

Appendix **1**

A.1 Extended Material

This extended material contains the details of the router configurations made in the Ubuntu box and some results that obtained during the implementation carried out in a virtual box environment.

Following these Linux commands in steps would start the radvd in the Linux machine and keep it running to enable the router to send periodic RAs [6].

1.   INTERFACE SETUP

```
malicious@NEAR:~$ sudo vi /etc/network/interfaces
       # interfaces(5) file used by ifup(8) and ifdown(8)
       iface eth0 inet6 static
       address 2001:dbb:0:1::1
       netmask 64
```

`[ ok ]` `malicious@NEAR:~$ sudo /etc/init.d/networking restart`

2.   ENABLE IPv6 FORWARDING
```
malicious@NEAR:~$ sudo vi /etc/sysctl.conf
malicious@NEAR:~$ sudo sysctl -w net.ipv6.conf.all.forwarding=1
net.ipv6.conf.all.forwarding = 1
```
3.   ROUTE SET UP
```
malicious@NEAR:~$ sudo ip a a 2001:dbb:0:1::1 dev eth0
malicious@NEAR:~$ sudo ip r a 2001:dbb:0:1::/64 dev eth0
```
4.   INSTALL WIRESHARK

```
malicious@NEAR:~$ sudo apt-get install wireshark
```
Start Wireshark

5.   INSTALL RADVD AND CONFIGURE

```
malicious@NEAR:~$ sudo apt-get install radvd
```

```
interface eth0
{
  AdvSendAdvert on;
  prefix 2001:dbb:0:1::/64
   {
      AdvOnLink on;
      AdvAutonomous on;
   };
};
malicious@NEAR:~$ sudo /etc/init.d/radvd restart
Stopping radvd: radvd.
Starting radvd: radvd.
```

Analyze the packets for the different parameters.

If the radvd does not start because no link local address is configured, the interface would be down and need to be brought up as shown in Fig.A.1.

```
malicious@NEAR:~$ sudo /etc/init.d/radvd restart
Stopping radvd: radvd.
Starting radvd: [Jul 20 14:06:05] radvd: no linklocal address configured for eth0
radvd.
malicious@NEAR:~$ sudo ifup eth0
```

Fig.A.1. Bringing up interface to configure the link local address at the router

After the router is setup to start radvd, we can observe the router advertisements send periodically from router using *radvdump* command at the router terminal. The *radvdump* is used to visualize the radvd configurations at the router while it sends out a RA as shown in Fig.A.2. The variables in the Fig.A.2 are explained in chapter 2.

The Ubuntu router is connected to the Comtrend Access point via its *eth0* interface as shown Fig.A.3. The Comtrend AP is configured using its IP address 192.168.1.1 to enable IPv6 traffic to go across it. The wireless devices in the network namely Android MS, iOS iPad and Windows host will be connected to the access point (AP) via wireless links. Android MS is connected via its *wlan0* interface as shown in Fig.A.4. The iOS iPad is connected via its *en0* interface to the AP which is seen in Fig.A.5 while the Windows host is connected via its *wireless LAN Adapter Wi-Fi* as seen in Fig.A.6.

```
# radvd configuration generated by radvdump 1.9.1
# based on Router Advertisement from fe80::21a:4bff:fe61:1121
# received by interface eth0
#

interface eth0
{
        AdvSendAdvert on;
        # Note: {Min,Max}RtrAdvInterval cannot be obtained with radvdump
        AdvManagedFlag off;
        AdvOtherConfigFlag off;
        AdvReachableTime 0;
        AdvRetransTimer 0;
        AdvCurHopLimit 64;
        AdvDefaultLifetime 7200;
        AdvHomeAgentFlag off;
        AdvDefaultPreference medium;
        AdvSourceLLAddress on;

        prefix 2001:dbb:0:1::/64
        {
                AdvValidLifetime 86400;
                AdvPreferredLifetime 14400;
                AdvOnLink on;
                AdvAutonomous on;
                AdvRouterAddr off;
        }; # End of prefix definition

}; # End of interface definition
#
```

Fig.A.2. Interface definition obtained with radvdump at router.

```
router@routerHP:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:1a:4b:61:11:21
          inet addr:192.168.1.8  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::21a:4bff:fe61:1121/64 Scope:Link
          inet6 addr: 2001:dbb:0:1:21a:4bff:fe61:1121/64 Scope:Global
          inet6 addr: 2001:dbb:0:1:a41a:52c6:614e:5398/64 Scope:Global
          inet6 addr: 2001:dbb:0:1::1/64 Scope:Global
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:218700 errors:0 dropped:0 overruns:0 frame:0
          TX packets:114182 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:319340264 (319.3 MB)  TX bytes:8477011 (8.4 MB)
          Interrupt:18
```

Fig.A.3. Network interface configuration of Router

Fig.A.4. Network interface configuration of Android MS

Fig.A.5. Network interface configuration of iOS IPad

Fig.A.6. Network interface configuration of Windows host

The following Fig A.7 shows the network set up of virtual box.
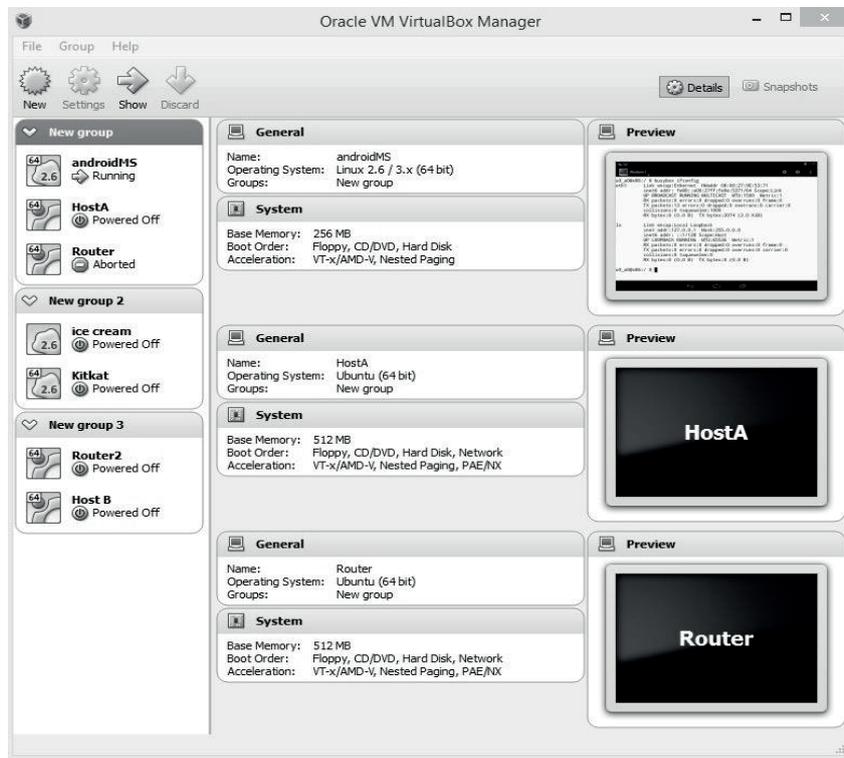
Fig.A.7. Virtual Box set up of the network for the analysis

The Fig.A.8, Fig.A.9, Fig.A.10 are the interface configurations of MS, Linux host and the Linux Router.

Fig.A.8. Network interface configuration of Android MS in Virtual Box

Fig.A.9. Network interface configuration of Router in Virtual Box

Fig. A.10. Network interface configuration of Router in Virtual Box

The Fig.A.11, Fig.A.12, Fig.A.13 are the RS, RA,NS,DAD and MLDv2 messages of MS, Linux host and the Linux Router.

Fig.A.11. Router Solicitation/Router Advertisement messages at the connection setup
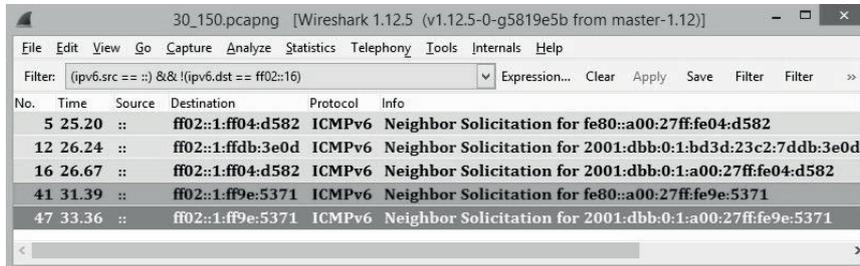
Fig. A.12. Duplicate Address Detection for the Linux host and Android MS
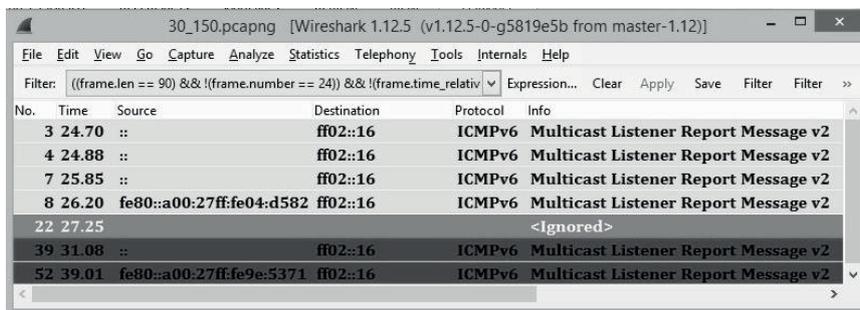
Fig.A.13. Multicast Listener Report v2 for Linux host and Android MS

The Fig.A.14 shows the messages exchanged during connection set up of MS and Linux Host.
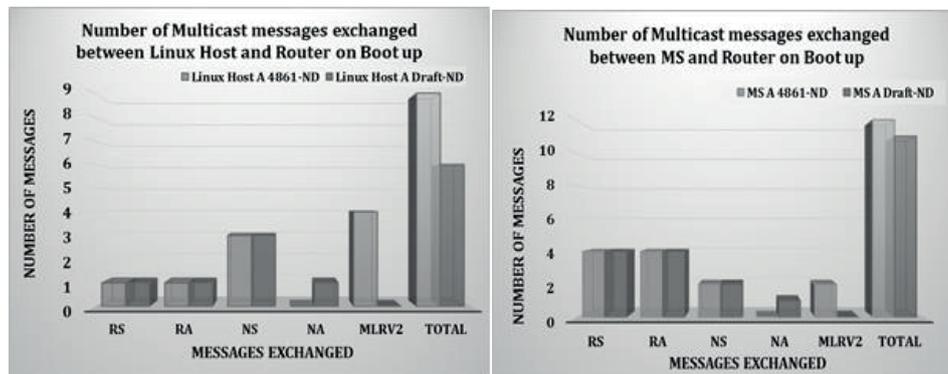

Fig.A.14. Messages exchanged during connection set up for Linux host and Android MS

The Fig. A.15 and Fig.A.16 shows the comparison of total RA messages and multicast messages in the network as per RFC 4861-ND for different time durations with that of expected results of efficient ND.

Fig.A.15. Comparison of total RA messages in the network as per RFC 4861-ND and expected number of RA messages as per Draft-ND for different time durations.


Fig.A.16. Comparison of total multicast messages in the network as per RFC 4861-ND and expected number of messages as per Draft-ND for different time durations.