Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on September 6, 2006.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

Mobile Ad Hoc Networks need an auto-configuration protocol being able to satisfy their self-deployment requirements while remaining flexible enough to accommodate special features for different devices.

The Extensible MANET Auto-configuration Protocol (EMAP) tries to fulfill these requirements both for IPv4 and IPv6 mobile ad hoc networks. It provides auto-configuration mechanisms for isolated as
well as hybrid MANETs, and is envisioned to be integrated within unicast routing protocols (like DYMO [1] or OLSRv2 [2]).

EMAP allows nodes to create a (highly likely) unique IP address which can be used locally inside the MANET. In a similar way, nodes can auto-configure globally routable IP addresses when one (or more) gateways to the Internet are present in the network. The general framework provided by the protocol may be used as a service discovery protocol for MANETs. As an example, this document also specifies an optional feature which consists on the discovery of DNS servers reachable from the MANET. However, the approach is extensible to other services like SIP proxies, authentication entities, etc. Therefore, EMAP has been designed taking into consideration the possibility of extending it later on with new features, uses, and optimizations.
Table of Contents

1. Introduction ............................................. 4
2. Terminology ............................................. 6
3. Protocol Overview ........................................ 7
   3.1. General EMAP message format ...................... 7
4. Local Configuration ...................................... 10
   4.1. Overview ........................................... 10
   4.2. Messages Format .................................... 10
      4.2.1. DAD_REQ (Duplicate Address Detection Request) .... 10
      4.2.2. DAD_REP (Duplicate Address Detection Reply) .... 10
   4.3. Protocol operation .................................. 11
5. Global Configuration ..................................... 13
   5.1. Overview ........................................... 13
   5.2. Messages Format .................................... 13
      5.2.1. GC_REQ (Global Configuration Request) ........... 13
      5.2.2. GC_REP (Global Configuration Reply) ............ 14
   5.3. Protocol operation .................................. 14
      5.3.1. IGW operation .................................. 14
      5.3.2. MANET nodes operation ......................... 15
   5.4. Comments about proxying in Global Configuration .... 17
   5.5. Adaptive Algorithm for Control Advertisements
        Dissemination ....................................... 17
6. DNS Server Configuration .................................. 19
   6.1. Overview ........................................... 19
   6.2. Messages Format .................................... 19
      6.2.1. DS_REQ (DNS Server Request) ................... 19
      6.2.2. DS_REP (DNS Server Reply) ..................... 19
   6.3. Protocol operation .................................. 19
7. Constants Values ......................................... 21
8. IANA Considerations ...................................... 22
9. Security Considerations .................................. 23
10. Acknowledgements ....................................... 24
11. References ............................................. 24
    Authors’ Addresses ...................................... 26
Intellectual Property and Copyright Statements ............ 27
1. Introduction

Due to the spontaneous nature of Mobile Ad Hoc Networks (MANET), a mechanism is required to automatically provide some basic configuration information to MANET nodes, allowing them to establish communications.

First of all, a MANET node needs an IP address that can be used locally for intra-MANET connectivity. One of the most important features of a MANET is that it does not need any pre-existing infrastructure, but MANET routing protocols assume unique addressability of all nodes. Currently there is no widely accepted mechanism for MANETs to provide every node with a unique IP address (other than manual configuration, which is, indeed, a pre-existing infrastructure). This document tries to fill this gap.

Hybrid MANETs may be permanently or temporarily attached to the Internet through one or more Internet Gateways. Every node needs a unique global address and a route to the Internet if it wants to communicate with external nodes.

In addition, there are other network parameters such as addresses of DNS servers which can be self-configured by a MANET node.

This document describes an auto-configuration protocol called EMAP which is able to provide all above mentioned services. It has been designed taking into consideration its extensibility as one of its most important features, so that it can be extended with additional functionality in the future. Although it MAY be implemented as a standalone daemon, EMAP SHOULD be implemented within an ad hoc routing protocol because it needs to create routes to other nodes, which is a routing protocol task.

To achieve the aforementioned extensibility, the general packet format defined in [15] has been adopted. It allows external and internal extensibility; the former by easily adding new message types, and the latter by appending new attributes to the existing messages.

Furthermore, EMAP is a flexible protocol designed to accommodate requirements for several devices. The current specification considers local and global auto-configuration as mandatory features, but the discovery of DNS servers is optional. Future extensions will include the discovery of new optional services.

Previous solutions have primarily focused on either local [3] [4] or global auto-configuration [5] [6]. Besides, some are only related to IPv6 [5][6]. EMAP extends the ideas of the previous proposals to
design a new protocol which deals both with local and global configuration and with IPv4 and IPv6.

The framework defined by this protocol can be used as a service discovery mechanism. Those elements which provide a service (i.e., servers) can periodically advertise their presence to the network, or may be silently waiting for the clients of the service to initiate a request-reply procedure. EMAP allows proxies (i.e., intermediate nodes) to reply to requests for which they know the suitable information. The latter is actually similar to the way AODV [7] behaves when a node wants to discover a route to another.

Both the use of proxies and the proposal of an adaptive algorithm to disseminate protocol information to the ad hoc network (Section 5.5), are aimed at the reduction of the protocol overhead. It is important to not overuse the scarce resources of the ad hoc network.

Local configuration is achieved by allowing a MANET node to choose an IP address and to ask the network if it is already being used (in order to avoid address duplication). If not, then it can begin to use that address. On the other hand, global configuration needs the presence of an element called an Internet Gateway which is responsible for providing suitable information to MANET nodes. This information is used to configure a unique global address and a route towards the Internet. In the last (but optional) service described within this document, a MANET node may issue a query to find DNS Server Advertisers, which provide IP addresses of available DNS servers.
2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", 
"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this 
document are to be interpreted as described in RFC-2119 [8].

All terminology defined in [9] also applies. Here are some additional terms used within this document.

DNS Server Advertiser

A device which knows the addresses of DNS servers and is responsible for providing that information to every MANET node which requests it.

Proxy

A MANET node which is not providing a concrete service but has enough information to answer requests for that service.

Temporary address

IP address used by a node for certain protocol operations before it acquires a unique IP address for intra-MANET or global communications.

Tentative address

Selected IP address for intra-MANET or global communications before the uniqueness check is performed.

MANET-local address

IP address to be used for intra-MANET communications. The scope and format of this sort of addresses is still an open issue.
3. Protocol Overview

EMAP defines a simple mechanism to request network configuration parameters in an efficient way, based on the use of Request and Reply messages.

When a node wants to auto-configure some sort of information it floods a request and waits for the network to reply. In the case that a response arrives, the next time that the node needs that same information it MAY send in unicast (or flood) a request to the node(s) which previously replied. In a similar way, a node MAY flood (periodically or not) a reply throughout the network if it wants to announce the existence of some type of information to the whole MANET, or it MAY wait for requests and then reply in unicast to only those requesting nodes.

Sometimes a node receives a request for some information which is not being provided by itself, but it knows the requested information and the node which provides it. In those cases the node MAY act as a proxy and reply to the originator. Then the request is not forwarded any more and the protocol overhead may be reduced.

Every time a node floods an EMAP message, it SHOULD use whatever optimized mechanism which is available (e.g. using a forwarding algorithm based on Connected Dominating Sets). This draft could have delegated the forwarding responsibility to SMF [10], but this presents a problem with the use of proxies because there is no such concept in that specification.

In order to avoid the re-processing and re-forwarding of duplicated messages, this protocol caches some information of the received messages. Then when a new message arrives a node can check if it is a duplicate and then discard it. An alternative option could be the use of the Duplicate Packet Detection (DPD) mechanism described in SMF spec [10], and probably future versions of this draft could use it.

3.1. General EMAP message format

Every EMAP message MUST conform to the generic message format described in [15]. For the sake of completeness, an outline of the message format is provided here:
<message> = <msg-header>
   <tlv-block>
   {<addr-block><tlv-block>}*

<msg-header> = <type>
   <msg-semantics>
   <msg-size>
   <msg-header-info>

<msg-header-info> = <originator-address>?
   <ttl>?
   <hop-count>?
   <msg-seq-number>?

<tlv-block> = <tlv-length>
   <tlv>*

<address-block> = <head-length>
   <head>
   <num-tails>
   <tail>*

<tlv> = <type>
   <tlv-semantics>
   <length>?
   <index-start>?
   <index-stop>?
   <value>?

All EMAP messages share the following features:

- Currently this protocol only specifies EMAP_REQUEST and EMAP_REPLY types for <msg-header> <type> field.

- Code TLV (a message TLV) MUST exist. It indicates the configuration which is being performed (currently local, global and DNS servers auto-configuration).

- <msg-semantics> MUST contain the P bit. If set in a request, this flag indicates that receivers SHOULD act as proxies. If not set they MUST NOT act as proxies. If set in a reply, it means that this message has been generated by a proxy.

- All fields contained in <msg-header-info> MUST exist, so that <msg-semantics> MUST be set accordingly.

- <originator-address> is the IP(v4/v6) address of the node that originated the information contained in this message. In the case
of replies which have been proxied (P bit set to 1), this field contains the address of the node which originally provided the information (i.e., not the proxy).

- If the P-bit is set in an EMAP_REPLY, <address-block> MUST contain one and only one address marked as the Proxy. If the P-bit is unset or this is an EMAP_REQUEST, the message MUST NOT have any Proxy TLV.

Besides these characteristics, every EMAP message has its own specific fields as we will see in the following sections.

Flooded EMAP messages are sent to the neighboring nodes. They will decide if an EMAP message is going to be forwarded or not (what is called "application flooding" [11]). The transmission range of a flooded EMAP message is controlled by the <ttl> field in the EMAP message.

Every time a node forwards an EMAP message, it MUST decrease its <ttl> field by one unit. When <ttl> reaches 0, the message MUST NOT be forwarded any more.

The following sections describe how to perform local and global address auto-configuration, as well as a mechanism to discover available DNS servers.
4. Local Configuration

4.1. Overview

Local Configuration allows a MANET node to communicate with other nodes in the same MANET. To achieve this, the node MUST allocate a unicast IP address which is unique in the connected part of the MANET. Due to the decentralized nature of MANETs, that address will be auto-configured in a stateless fashion.

A MANET node which wants to auto-configure an address for local use, picks an IP address and asks for it to the rest of the network (sending a DAD_REQ message). If that IP address is already in use by another node, then the network replies with a DAD_REP message indicating that the node cannot auto-assign that address itself. If the IP address is not being used by any node, the network does not answer and the node can use it.

The latter mechanism is a Duplicate Address Detection procedure for MANETs, (MANET-DAD), used to guarantee the uniqueness of self-configured IP addresses. The main idea has been taken from [3], and proxying capability has been added to the nodes’ functionality in order to lower the control overhead when the IP address being requested is already assigned to other node.

Subsequent sections give more insight on the protocol for Local Configuration and explain it in detail.

4.2. Messages Format

4.2.1. DAD_REQ (Duplicate Address Detection Request)

A DAD_REQ message MUST set the message <type> field to EMAP_REQUEST and the Code TLV <value> to DAD_CODE. It also has one specific field:

Requested Address. IP(v4/v6) address which the node is requesting. This MUST be the selected "tentative address".  

<address-block> MUST contain one and only one address marked with the Requested Address TLV.

4.2.2. DAD_REP (Duplicate Address Detection Reply)

A DAD_REP message MUST set the message <type> field to EMAP_REPLY and the Code TLV <value> to DAD_CODE. It has no specific message fields.
4.3. Protocol operation

Every EMAP-compliant implementation MUST implement Local Configuration.

When a node wants to join a MANET it needs a valid and locally unique IP address for (at least) one of its interfaces. The following procedure MUST be applied on each interface interested in participating in intra-MANET communications when there is no suitable address which could be used (e.g. a Mobile IP Home Address). If such an address exists, the node MAY optionally execute the procedure.

A node generates a pair of IP addresses, namely the temporary and the tentative ones. The temporary address is used as the <originator-address>, and it can be a Mobile IP Home Address or other sort of (highly likely) unique address. If there is no such address available then the temporary address is automatically created. On the other hand, the tentative address is the one which is being requested and is used as the Requested Address field in DAD_REQ and <originator-address> field in DAD_REP messages.

In the case of IPv4, the node creates a temporary and a tentative MANET-local addresses from the 169.254/16 subnet. For the temporary address, the 16 low-order bits are generated by randomly choosing a number in the range 1 - 2047. Later on the node randomly picks a number from the range 2048 - 65534 which is used in the 16 low-order bits to create the tentative address. This mechanism may require more discussion within the IETF and it can therefore be changed in future versions of this draft.

Similarly, in the case of IPv6 there is no consensus yet. A suitable solution is to use Unique Local Addresses with an special Global ID field reserved for MANET use as MANET-local addresses. Subnet ID is randomly chosen in the range 1 - 2047 for the temporary address, and in the range 2048 - 65534 for the tentative address. Interface ID is set to the EUI of the interface which is requesting the tentative address.

A requesting node issues a DAD_REQ message, where the <originator-address> is the temporary address and the Requested Address is the tentative address. It then waits DAD_REQ_TIMEOUT seconds for a DAD_REP message. If this timer expires without receiving any answer the requesting node assumes the uniqueness of the tentative address, deallocates the temporary address and assigns the tentative address to the corresponding interface. On the other hand if the node receives a DAD_REP destined to its temporary address where the <originator-address> is the tentative address, then the latter is being used by another node in the MANET and cannot be auto-assigned.
In this case the requesting node MUST re-run the whole process until it succeeds or DAD_MAX_RETRIES retries are reached.

Before sending the DAD_REQ message, the node MAY set the P bit to indicate that intermediate nodes SHOULD answer with a DAD_REP message if they do not own the Requested Address but they do know that it is being used by another node. For instance, if the MANET is running a proactive routing protocol then every node should have a route to every other node. In that case a node can easily check if it knows about another node using a given address. On the other hand, if a reactive protocol is being run then a node could know if an address is already assigned if that node is part of the path of an active communication in which that address is the source or destination.

When a node receives a DAD_REQ message it MUST check whether there exists an entry in the DAD_REQ_CACHE with the <originator-address> and the Requested Address of this message. If that is the case, then the message has been already processed and MUST be discarded. In any other case, a new entry which expires after DAD_REQ_CACHE_TIMEOUT seconds is created in the DAD_REQ_CACHE. The routing protocol being run (reactive or proactive) MUST create a reverse route entry in its routing table where the destination address is the <originator-address> and the next hop is the node from which it received the DAD_REQ message. This is needed to forward the eventual DAD_REP message.

After that, the node checks if the Requested Address is owned by one of its interfaces. In that case, a new DAD_REP message is issued in unicast directed to the <originator-address>. In other case if the P bit is active and the Requested Address is not owned by the node but it knows the address is already being used, then a new DAD_REP with the P flag activated SHOULD be sent in unicast to the <originator-address>. If the P bit is not active in the request, then the node MUST NOT act as a proxy and therefore MUST NOT reply with a DAD_REP message. In other case, (i.e., if the Requested Address does not belong to the receiving node and it cannot act as a proxy) then the node MUST forward the DAD_REQ message if the <ttl> field is bigger than zero.
5. Global Configuration

5.1. Overview

Global Configuration allows MANET nodes to communicate with others in the Internet. For this purpose a MANET node needs at least one globally valid IP address. Due to the hierarchical structure of the Internet that address must be globally routable.

To reach the Internet at least one element must act as a gateway between the MANET and the fixed network. This is called an Internet-Gateway (IGW). An IGW may be a fixed element belonging to each network, or a mobile one which detects the presence of an attachment point to Internet (e.g. a wireless router).

IGWs are responsible for announcing suitable network parameters which will be used by MANET nodes in order to:

1. Auto-configure in a stateless fashion a globally routable and unique IP address.

2. Discover (at least) a path to route packets destined to the nodes located in the Internet.

Information supplied to MANET nodes by IGWs is detailed in the following sections. This information MAY be delivered proactively, reactively, with a hybrid scheme or MAY be even dynamically changed depending on the implementation and network operator preferences. Later on, Section 5.5 describes an adaptive algorithm suggested by this specification.

There are several different ways to route traffic to the Internet, once IGWs are known. E.g. the routing protocol could tunnel the traffic to the IGW, use IPv6 Routing Headers, create default routes, etc. These options are not discussed within this document.

The main idea of the protocol operation has been taken from [5], and proxying capability and a new adaptive algorithm have been added.

5.2. Messages Format

5.2.1. GC_REQ (Global Configuration Request)

A GC_REQ message MUST set the message <type> to EMAP_REQUEST and the Code TLV <value> to GC_CODE. It has no additional fields.
5.2.2. GC_REP (Global Configuration Reply)

A GC_REQ message MUST set the message <type> to EMAP_REPLY and the Code TLV <value> to GC_CODE. It also has the following specific fields:

- **Lifetime.** Indicates the time in seconds the information contained in this message is considered valid. This field is expressed in a mantissa/exponent format like it is defined in the OLSR protocol specification [14]. The Lifetime TLV (message TLV) MUST exist.

- **Prefix Length.** Length (in bits) of the prefix being advertised by the IGW. The Prefix Length TLV (message TLV) MUST exist.

- **S bit.** Selected bit, if set the IGW announced in this message has been selected by the previous hop in order to create/update its default route. <msg-semantics> MUST contain the P bit.

5.3. Protocol operation

Every EMAP-compliant implementation MUST implement Global Configuration.

5.3.1. IGW operation

Concrete behaviour of IGWs depends on their particular implementation and configuration. They MAY proactively flood GC_REP messages. Or they MAY answer in unicast to every GC_REQ message they receive. Another option is to flood periodically GC_REP messages to the nearer MANET nodes, and let the farthest ones operate in an on-demand basis. It is even possible to dynamically adapt the reactivity/proactiveness behaviour of the IGWs as in Section 5.5. In the case that EMAP is integrated within a proactive routing protocol, IGWs SHOULD also operate in a proactive way. If it is integrated within a reactive routing protocol, we RECOMMEND to use the adaptive algorithm described in Section 5.5.

In all cases an IGW MUST send a GC_REP message when it receives a GC_REQ message. This is sent in unicast to the <originator-address> of the GC_REQ message.

Every IGW MUST set the S bit in each GC_REP it generates, and MUST NOT set the P bit unless it is proxying another IGW. Besides, each IGW maintains an internal sequence number counter which is incremented every time a new GC_REP is sent. This counter is used to fill the <msg-seq-number> field of the GC_REP message. <originator-address> field is set to the IGW IP address, <hop-count> field to 0, and Lifetime TLV <value> and <ttl> fields are set according to
implementation preferences.

When an IGW receives a DAD_REQ message, it MAY answer with a GC_REP message directed in unicast to the <originator-address>. Hence, the originating node is able to auto-configure a global address by substituting the first bits of the requested local address by the prefix advertised by the IGW.

5.3.2. MANET nodes operation

There are two ways a MANET node may acquire global connectivity. If the IGW is proactively sending GC_REP messages, the node MUST process those messages. On the other hand, if the node needs global auto-configuration at a certain time and it has not received any GC_REP, then it MUST flood a GC_REQ message.

If the node previously performed Local Configuration, then the acquired MANET-local address SHOULD be used as the <originator-address>. Otherwise, the node MAY use other available address such as a Mobile IP Home Address. If there is no such available address then the node MUST use a temporary address obtained in the same way as in Local Configuration. In fact, if the node is requesting both local and global auto-configuration at the same time then the same temporary address SHOULD be used.

When a MANET node receives a GC_REQ message it MUST check whether there exists an entry in the GC_REQ_CACHE with the <originator-address> of this message. If that is the case, then the message has been already processed and MUST be discarded. In other case, a new entry which expires after GC_REQ_CACHE_TIMEOUT seconds is created in the GC_REQ_CACHE.

Similarly, when a MANET node receives a GC_REP message it MUST check if this is an old message looking into GC_REP_CACHE. If there exists an entry with the IGW address given in the GC_REP and with a higher or equal <msg-seq-number> then the message is discarded (and the processing ends here). Else the entry is updated or newly created.

When a MANET node processes a GC_REP message, it SHOULD create (and allocate) a new global address with the information contained in that message. If the node does not have a global address yet, then it MUST create (and allocate) that address.

In the case of IPv4, the node creates a tentative address by appending a random number to the network subnet obtained from the <originator-address> and the Prefix Length fields. In order to guarantee the uniqueness of the tentative address, a MANET-DAD procedure SHOULD be performed before allocating that address to an
In the case of IPv6, if the length of the prefix advertised by the IGW is lower than 64 bits then a random number is generated in order to complete the 64 high-order bits, and the EUI of the interface is added to get the entire address. When the tentative address is auto-configured on this way, the node MAY perform a MANET-DAD procedure to assure the uniqueness of the tentative address. Otherwise the address is completed by adding a random number to the prefix advertised by the IGW. In this latter case the node SHOULD perform the MANET-DAD procedure because of the higher likelihood of address collision.

The MANET-DAD procedure mentioned in the two paragraphs above is performed in the same way as in Local Configuration, by using DAD_REQ and DAD_REP messages.

Every time a GC_REP is received from the same IGW the auto-configured address with its information is refreshed. That is, its lifetime is reset to the value inside Lifetime TLV <value>. If this timer expires and the node is communicating with nodes outside the MANET using that address as the source, then a new GC_REQ is issued in unicast to the corresponding IGW. Else if it is not being used, then no GC_REQ is sent. In the first case, if the node sent a GC_REQ in unicast but no response was received, then it MUST flood again the GC_REQ.

When there are multiple IGWs announcing their own information, the MANET node SHOULD select one in order to create a default route to the Internet. Which rules are followed to select this IGW are implementation-dependent, but they will likely involve Distance field of the GC_REP message. Future extensions to this protocol may define new fields with information which can be used to select the appropriate default IGW.

When a MANET node routes traffic to the Internet via an IGW, it SHOULD use the auto-configured global address which has been obtained from that IGW as the source address of every IP Datagram (to avoid ingress filtering).

After processing a GC_REP message which is being flooded or sent in unicast to another node, the MANET node MUST forward a new version of the message. This new message is like the original GC_REP but with the following changes:

1. New <hop-count> := Old <hop-count> + 1
2. $S := 1$ if this message has been used to create/refresh the default route entry. $S := 0$ otherwise.

5.4. Comments about proxying in Global Configuration

There are several issues arising when we allow a MANET node to reply instead of the IGW.

First of all, information provided by the proxy may be not fully fresh compared to the one provided by the IGW. This may cause a bigger latency since a node starts using a global IP address and a route towards Internet to the time it discovers the IGW is no longer available (e.g. if the IGW has been shut down).

In addition, when there are multiple IGWs in the network the proxy can take two different approaches: send as many GC_REP messages as IGWs it knows, or only send the one which refers to the IGW which it used to create its default route. The former increases the overhead of the protocol, and the latter only provides partial information about the IGWs present in the network.

In spite of these troubles, proxies may be useful in some scenarios. As an example, think of a large MANET running a reactive routing protocol and an IGW attached on an edge. The IGW could proactively send GC_REPs messages to the nearest nodes in order to provide them with fast and updated IGW information. Then a node located further away could send a GC_REQ with the P bit active, trying to minimize the protocol overhead because that message will not need to be flooded throughout the whole network.

Future versions of this draft will try to give a better understanding of the trade-offs involved in the use of proxies.

5.5. Adaptive Algorithm for Control Advertisements Dissemination

In this section we detail a simple algorithm which can be used by IGWs to disseminate protocol messages through the ad hoc network. This is intended to be used with GC_REP messages, although other message types could be propagated by using this same algorithm.

We expect the reactive discovery of IGWs to poorly scale with regard to the number of sources, since every route discovery provokes the sending of a multicast message to the whole ad hoc network. However, that approach may achieve a great packet delivery ratio since a route is looked for as soon as it is needed. So, packets do not tend to be dropped because queues get full. Moreover, this solution scales well when the number of IGWs present in the network increases.
On the other hand, the proactive approach has got the opposite characteristics. The protocol overhead is the same regardless the number of route discoveries that need to be performed. But it heavily gets higher when there are many IGWs available. Depending on the rate of the advertisements (GC_REPs) sent out by the IGWs, the ad hoc nodes may wait too much time to get the information they need before sending their data packets, making queues to get full and drop packets.

To find a trade-off between both schemes, hybrid solutions may be adopted. IGWs can advertise their control messages to the nearest nodes, and let the farthest ones to operate on an on-demand basis. The problem here is to determine which is the appropriate <ttl>. We propose an adaptive TTL scoping based on the distance to the sources which are using a given IGW. Several algorithms may be used, but here we opt for one of the simplest: the maximal source coverage.

To implement this algorithm, IGWs must know the number of hops from them to every source. This information may be hard to acquire from the data packets, but here we depict some ways:

- In the easiest way, the IGW may assume a fixed default TTL for every source, or make a guess for every one. Then the number of hops may be got by subtracting the TTL/Hop Limit field of the IP header from the guessed default TTL. Obviously, this method gives us just an approximation but is easy to implement.

- Another option is to include in the request messages the default TTL that is used by the source. So, the exact number of hops can be computed every time a data packet is received.

- Finally, a new Option Header for IPv4/v6 may be included in data packets sent by ad hoc sources. This new header would include the original TTL/Hop Count which was used when the packet was created.

Once the IGW knows the number of hops between itself and every source which use it to communicate to the Internet, it may dynamically adapt the <ttl> scope of its advertisements to the number of hops to the farthest source.

A description and study of the performance of this algorithm and a more elaborated one can be found in [13].
6. DNS Server Configuration

6.1. Overview

DNS Server Configuration allows MANET nodes to discover DNS servers which are available to the network. An special node called DNS Servers Advertiser (DSA) will provide the IP addresses of a primary and perhaps a secondary DNS server. With this information every MANET node can configure itself to issue DNS requests towards those DNS servers. This feature may be quite useful in situations where it is desired a high degree of auto-configuration.

DSAs are not necessarily independent devices but they may be integrated within IGWs.

6.2. Messages Format

6.2.1. DS_REQ (DNS Server Request)

A DS_REQ message MUST set the message <type> to EMAP_REQUEST and the Code TLV <value> to DS_CODE. It has no additional fields.

6.2.2. DS_REP (DNS Server Reply)

A DS_REP message MUST set the message <type> to EMAP_REPLY and the Code TLV <value> to DS_CODE. It carries the following additional information:

   Lifetime. Indicates the time in seconds the information contained in this message is considered valid. This field is expressed in a mantissa/exponent format like it is defined in the OLSR protocol specification [14]. The Lifetime TLV (message TLV) MUST exist.

   Primary DNS Server Address. IP(v4/v6) address of the advertised primary DNS server. <address-block> MUST contain one and only one address marked with the Primary DNS TLV.

   Secondary DNS Server Address. IP(v4/v6) address of the advertised secondary DNS server. <address-block> MAY contain one or more addresses marked with the Secondary DNS TLV.

6.3. Protocol operation

An EMAP-compliant implementation MAY implement DNS Server Configuration.

When a MANET node wants to discover DNS servers which are near the ad hoc network, then it floods a DS_REQ message trying to find a DSA. A
Internet-Draft  EMAP  March 2006

DSA MAY also flood DS_REP messages regularly; following either a proactive, hybrid or adaptive approach.

If a MANET node receives a DS_REQ message it MUST check whether there exists an entry in the DS_REQ_CACHE with the <originator-address> of this message. If that is the case, then the message has been already processed and MUST be discarded. In other case, a new entry is created in the DS_REQ_CACHE. This entry will expire after DS_REQ_CACHE_TIMEOUT seconds.

As the <originator-address> the node can use any of the addresses we have talked before in this document: MANET-local address, stateless global address, any fixed global address such as the Mobile IP Home Address, or a temporary address computed in the same way than in previous services. It MAY activate the P bit if it allows proxies to reply in the name of a DSA.

When a DSA receives a DS_REQ then it MUST reply in unicast to the <originator-address> with the IP addresses of a primary and zero or more secondary DNS servers. If the DS_REQ had the P bit active, then a MANET node which is acting as a DSA proxy SHOULD also answer with a DS_REP directed in unicast to the <originator-address>. In this latter case, the <originator-address> field is set to the DSA actual IP address.

At the time a MANET node receives the DS_REP message that it was waiting for, then it can start using the DNS servers which have been advertised.
## 7. Constants Values

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAD_REQ_TIMEOUT</td>
<td>TBD</td>
</tr>
<tr>
<td>DAD_REQ_CACHE_TIMEOUT</td>
<td>TBD</td>
</tr>
<tr>
<td>DAD_MAX_RETRIES</td>
<td>5</td>
</tr>
<tr>
<td>GC_REQ_CACHE_TIMEOUT</td>
<td>TBD</td>
</tr>
<tr>
<td>GD_REQ_CACHE_TIMEOUT</td>
<td>TBD</td>
</tr>
<tr>
<td>DS_REQ_CACHE_TIMEOUT</td>
<td>TBD</td>
</tr>
</tbody>
</table>
8. IANA Considerations

EMAP defines several `<message-types>` and `<tlv-types>`. A new registry should be created for the values of the various type fields, with the following values:

```
+--------------+-------+
| Types        | Value |
+--------------+-------+
| EMAP_REQUEST | TBD   |
| EMAP_REPLY   | TBD   |
| DAD_CODE     | TBD   |
| GC_CODE      | TBD   |
| GD_CODE      | TBD   |
| DS_CODE      | TBD   |
| CODE_TLV     | TBD   |
| PROXY_TLV    | TBD   |
| REQADDR_TLV  | TBD   |
| LIFETIME_TLV | TBD   |
| PRELEN_TLV   | TBD   |
| PRIDNS_TLV   | TBD   |
| SECDNS_TLV   | TBD   |
+--------------+-------+
```
9. Security Considerations

This document does not define any method for secure operation of the protocol. It will be addressed in future versions.
10. Acknowledgements

Many of the ideas of this protocol have been elaborated thanks to the live and interesting discussions from the MANET and MANET-AUTOCONF mailing lists. We would also like to thank Thomas Clausen, Shubhranshu Singh and Charles Perkins for their comments on previous versions of this draft.

11. References


February 2005.


Authors’ Addresses

Francisco J. Ros
University of Murcia
Facultad de Informatica
Campus de Espinardo
Espinardo, Murcia  E-30100
Spain

Phone: +34 968 367938
Email: fjrm@dif.um.es

Pedro M. Ruiz
University of Murcia
Facultad de Informatica
Campus de Espinardo
Espinardo, Murcia  E-30100
Spain

Phone: +34 968 367646
Email: pedrom@dif.um.es
Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Copyright (C) The Internet Society (2006). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.