Network Working Group
Internet-Draft
Intended status: Experimental
Expires: October 9, 2008

A. Neumann
C. Aichele
M. Lindner
S. Wunderlich

Apr 07, 2008

Better Approach To Mobile Ad-hoc Networking (B.A.T.M.A.N.)
draft-wunderlich-openmesh-manet-routing-00

Status of this Memo

Copyright Notice

Abstract

This document specifies a simple and robust algorithm for establishing multi-hop routes in mobile ad-hoc networks. It ensures highly adaptive and loop-free routing while causing only low processing and traffic cost.

Table of Contents

1. Introduction 3
1.1. Terminology 4
1.2. Protocol Overview 4
2. Protocol and Port Number 6
3. B.A.T.M.A.N. Packet Formats 6
3.1. General B.A.T.M.A.N. Packet Format 6
3.2. Originator Message (OGM) Format 8
3.2.1. Originator Message (OGM) Fields 8
3.3. HNA Message Format 9
3.3.1. HNA Message Fields 9
4. Conceptual Data Structures 9
4.1. Originator List 9
4.2. Sequence Numbers, Ranges, and Windows 11
5. Flooding Mechanism 13
5.1. Broadcasting own Originator Messages (OGMs) 13
5.2. Receiving Originator Messages 14
5.3. Bidirectional Link Check 14
5.4. Neighbor Ranking 15
5.5. Re-broadcasting other nodes' OGMs 15
6. Routing 16
6.1. Route Selection and Establishing 16
6.2. Route Deletion 17
6.3. Opportunistic Routing Deletion Policy 17
6.3.1. Opportunistic Routing Deletion Policy Consideration . 17
7. Gateway 17
7.1. Gateway Announcement 17
7.2. Gateway Selection 18
7.3. Gateway Tunneling/Encapsulation 19
7.4. Gateway hopping (testing/accepting) 19
8. Proposed Values for Constants 19
9. IANA Considerations 20
10. Security Considerations 20
10.1. Confidentiality 20
10.2. Overflow of routing entries 20
10.3. Route manipulation 21
11. References 21
11.1. Normative References 21
11.2. Informative References 22
Appendix A. Contributors 22
Authors' Addresses 22
Intellectual Property and Copyright Statements 24

1. Introduction

B.A.T.M.A.N. is a proactive routing protocol for Wireless Ad-hoc Mesh Networks, including (but not limited to) Mobile Ad-hoc Networks (MANETs). The protocol proactively maintains information about the existence of all nodes in the mesh that are accessible via single-hop or multi-hop communication links. The strategy of B.A.T.M.A.N. is to determine for each destination in the mesh one single-hop neighbor, which can be utilized as best gateway to communicate with the destination node. In order to perform multi-hop IP-based routing, the routing table of a node must contain a link-local gateway for each host or network route. To learn about the best next-hop for each destination is all that the B.A.T.M.A.N. algorithm cares about. There is no need to find out or calculate the complete route, which makes a very fast and efficient implementation possible.

Wireless mesh networks have special difficulties unlike wired networks: Data packets can and will get lost in noisy areas. Mesh networks consisting of nodes with only one wireless communication interface (which are usually operating on the same wireless channel) have to cope with self-inflicted interference caused by their own wireless traffic. Thus communication links may have varying quality in terms of packet loss, data rates, and interference. Even the protocol traffic from the routing protocol itself causes interference. Therefore communication link quality changes even in static network topologies. New links appear and known links disappear frequently, especially in MANETs. The quality of one communication direction may differ to the opposite direction (e.g. asymmetric links).

B.A.T.M.A.N. considers these challenges by doing statistical analysis of protocol packet loss and propagation speed and does not depend on state or topology information from other nodes. Rather than trusting on metadata contained in received protocol traffic - which could be delayed, outdated, or lost - routing decisions are based on the knowledge about the existence or lack of information. B.A.T.M.A.N. protocol packets contain only a very limited amount of information and are therefore very small. Lost protocol packets due to unreliable links are not countered with redundancy, but are detected and utilized for better routing decisions. B.A.T.M.A.N. chooses the most reliable route upon the next-hop routing decision of individual nodes. This approach has shown in practise that it is reliable and loop-free.

Comments are solicited and should be addressed to the B.A.T.M.A.N. mailing list at b.a.t.m.a.n@open-mesh.net and/or the authors.

1.1. 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 [RFC2119].

Node - A mesh router which utilizes the B.A.T.M.A.N. protocol as specified in this document on at least one network interface.

Link - wired or wireless communication link, which can be unidirectional or bidirectional

Direct Link - single-hop communication link between two particular B.A.T.M.A.N. interfaces

Bidirectional Link - direct link with bidirectional (symmetric) communication capability

Unidirectional Link - direct link with only unidirectional (asymmetric) communication capability

Bidirectional Neighbor - single-hop neighbor, available via a direct bidirectional link

Best Link - the most promising outgoing interface and next hop towards a given originator

B.A.T.M.A.N. Interface - Network interface utilized by B.A.T.M.A.N. This is also a synonym for an Originator.

Host Network Announcement (HNA) - Message type, used to announce a gateway to a network or host

Originator Message (OGM) - B.A.T.M.A.N. protocol message advertising the existence of an Originator. OGMs are used for link quality and path detection.

Originator - Synonym for a B.A.T.M.A.N. Network Interface, announced by B.A.T.M.A.N. Originator Messages.

1.2. Protocol Overview

B.A.T.M.A.N. detects the presence of B.A.T.M.A.N.-Originators, no matter whether the communication path to/from an Originator is a single-hop or multi-hop communication link. The protocol does not try to find out the full routing path, instead it only learns which link-local neighbor is the best gateway to each Originator. It also keeps track of new Originators and informs its neighbors about their

existence. The protocol ensures that a route consists of bidirectional links only.

On a regular basis every node broadcasts an originator message (or OGM), thereby informing its link-local neighbors about its existence (first step). Link-local neighbors which are receiving the Originator messages are relaying them by rebroadcasting it, according to specific B.A.T.M.A.N. forwarding rules. The B.A.T.M.A.N. mesh network is therefore flooded with Originator messages. This flooding process will be performed by single-hop neighbors in the second step, by two-hop neighbors in the third step, and so forth. OGMs are send and repeated as UDP broadcasts, therefore OGMs are flooded until every node has received it at least once, or until they got lost due to packet loss of communication links, or until their TTL (time to live) value has expired. In practise OGM packet loss caused by interference, collision or congestion is significant. The number of OGMs received from a given Originator via each link-local neighbor is used to estimate the quality of a (single-hop or multi-hop) route. In order to be able to find the best route to a certain Originator, B.A.T.M.A.N counts the originator-messages received and logs which link-local neighbor relayed the message. Using this information B.A.T.M.A.N. maintains a table with the best link-local router towards every Originator on the network. By using a sequence number, embedded in each OGM, B.A.T.M.A.N. can distinguish between new Originator message packets and duplicates ensuring that every OGM gets only counted once.

B.A.T.M.A.N. was not designed to operate on stable and reliable media, such as cable networks, but rather to function on unreliable media inherently experiencing high levels of instability and data loss. The protocol was devised to counteract the side effects of a network’s fluctuation and compensate its instability, thus allowing for a high level of robustness. It also embodies the idea of collective intelligence opposed to link state routing. The topographical information is not handled by a single node, but spread across the whole network. No central entity knows all possible ways through the network. Every node only determines the data to choose the next hop, making the protocol very lightweight and quickly adapting to fluctuating network topologies.

B.A.T.M.A.N. Originators can announce themselves as gateways to the internet. Their announcement includes a gateway-class, which is specifying the connection speed of their up- and downlink to the internet. Gateways also send a port-number which is used by gateway clients to establish a unidirectional UDP-tunnel to the gateway. The decision which gateway is selected for a destination is performed by the gateway-client.

The method of tunneling between a B.A.T.M.A.N. internet gateway client and the internet gateway ensures a stable route to the internet as long as the protocol can maintain a working communication path between both peers. This is particularly important when the internet gateway has to perform Network Address Translation (NAT) between nodes using private IP address space in the mesh and public IP networks.

Once the tunnel is established the network topology and routing paths between the B.A.T.M.A.N. gateway and the gateway client may change but the data will get routed via the initial gateway and back without changes, as long as the protocol can provide a working communication route. Thus, B.A.T.M.A.N. is capable to provide stable session-based internet-traffic in MANETs with more than one gateway to other network segments. Apart from stable routing the tunneling also allows for techniques such as black hole detection to be used in B.A.T.M.A.N. networks.

2. Protocol and Port Number

Packets in B.A.T.M.A.N. are communicated using UDP. Port 4305 has been assigned by IANA for exclusive usage by the B.A.T.M.A.N. protocol.

3. B.A.T.M.A.N. Packet Formats

3.1. General B.A.T.M.A.N. Packet Format

The general layout of a B.A.T.M.A.N. packet (without the trailing IP and UDP header).

General B.A.T.M.A.N. packet format.

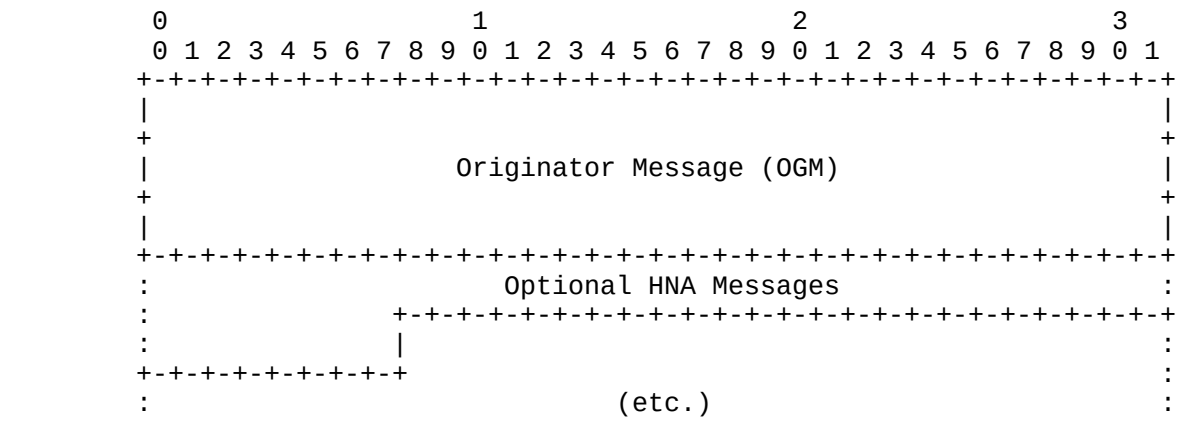

Figure 1

Each B.A.T.M.A.N. packet is encapsulated in a single UDP data packet.

A B.A.T.M.A.N. packet consists of one originator message (OGM) and zero or more attached HNA extension messages.

Originator Message (OGM):

> OGMs have a fixed size of 12 octets. They are further described in Section Section 3.2.

Optional HNA Extension Messages:

> An OGM may be followed by zero or more HNA extension messages. Each extension message following a preceding OGM is associated with the preceding OGM and MUST be processed respectively.
>
> HNA messages have a fixed size of 5 octets. It is described in Section Section 3.3.

3.2. Originator Message (OGM) Format

Originator Message (OGM) format.

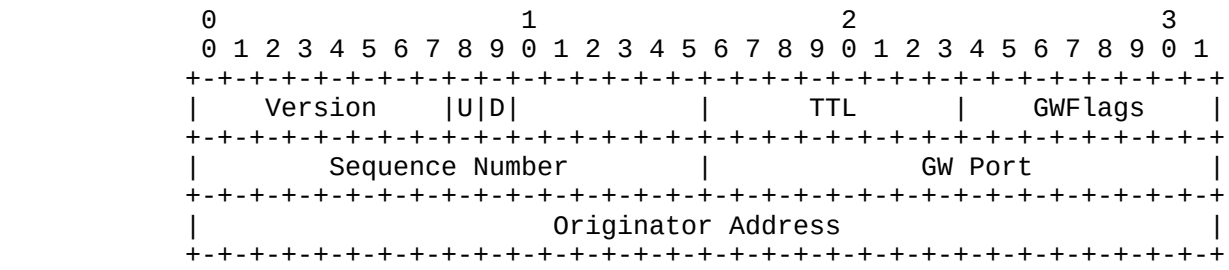

Figure 2

3.2.1. Originator Message (OGM) Fields

Version:

MUST be set to VERSION. Each packet received with a version field different from VERSION MUST be ignored.

Is-direct-link flag:

This flag indicates whether a Node is a direct neighbor or not.

Unidirectional flag:

This flag indicates whether the neighboring node is bidirectional or not.

TTL (Time To Live):

The TTL can be used to define an upper limit on the number of hops an OGM can be transmitted

Gateway flags (GWFlags):

MUST be set according to description in Section 7.1.

Sequence Number:

The originator of an OGM consecutively numbers each new OGM with an incremented (by one) sequence number. To get an overview about the Sequence Number handling see Section 4.2.

Originator Address:

The IPv4 address of the B.A.T.M.A.N. interface on which behalf the OGM has been generated.

3.3. HNA Message Format

HNA-extension-message format.

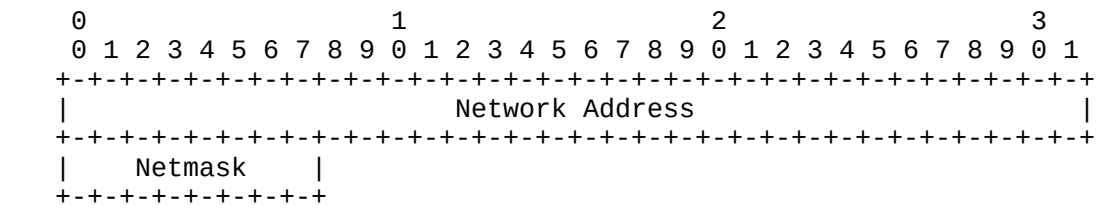

Figure 3

3.3.1. HNA Message Fields

Netmask:

The number of bits presenting the size of the announced network.

Network Address:

The IPv4 network address of the announced network.

4. Conceptual Data Structures

Each node must maintain certain information about its own and other B.A.T.M.A.N. originators in the network and how these originators are related to neighboring nodes and to the B.A.T.M.A.N. interfaces of the node itself. This Section conceptionally describes the information necessary to conform to the protocol described in this document.

4.1. Originator List

Each node maintains information about the known other originators (B.A.T.M.A.N. Interfaces) in the network in an Originator List. The Originator List contains one entry for each Originator from which or via which an OGM has been received within the last PURGE_TIMEOUT seconds. If OGMs from different Originators (B.A.T.M.A.N. interfaces) of the same node are received then there MUST be one

entry for each Originator. In fact, the receiving node does not necessarily know that certain different Originators (and corresponding IP addresses) are belonging to the same B.A.T.M.A.N. node.

For each Originator the following Information must be maintained:

- Originator IPv4 Address:

  The IPv4 address of the Originator (B.A.T.M.A.N. Interface) as given in the corresponding field of the received OGM.

- Last Aware time:

  A timestamp which MUST be updated with every OGM that has been received from or via the given Originator.

- Bidirect Link(s) Sequence Number:

  The bidirectional Link Check requires a Node to save the information which direct neighbor successfully rebroadcasted its own OGM. Therefore, the Sequence Number of the last accepted self-initiated OGM received from a direct link neighbor is to be saved here on a per Interface basis. This is described in Section 5.3.

- Current Sequence Number:

  The newest OGM Sequence Number that has been accepted from the given Originator. This is described in Section 4.2.

- HNA List:

  All announced Networks of the Originator with their IP-Range and Netmask.

- Gateway capabilities:

  If the Originator offers a Gateway, and its announced parameters.

- Neighbor information List:

  for each Direct Link to each Neighbor of the node the following information must be maintained:

  + Sliding Window:

For each In-Window Sequence Number it is remarked if an OGM with this Sequence Number has been received. This is described in Section 4.2.

+ Packet Count:

  The amount of Sequence Numbers recorded in the Sliding Window. It is used as a metric for the path to the Originator via this neighbor.

+ Last Valid Time:

  The timestamp when the last valid OGM was received via this neighbor.

+ Last TTL:

  The TTL of the last OGM which was received via this neighbor.

4.2. Sequence Numbers, Ranges, and Windows

B.A.T.M.A.N. is Sequence Number oriented. In fact, the Sequence Number of a received OGM is the key information that is transmitted with each OGM.

Sequence Numbers are recorded in dedicated sliding windows until they are considered Out-Of Range. Thus, such a sliding window always contains the set of recently received Sequence Numbers. The amount of Sequence Numbers recorded in the Sliding Window is used as a metric for the quality of detected links and paths.

The Sequence Number range is not an infinite space but is limited to the range of 0 .. 2^16 - 1. Since the space is limited, all arithmetical operations must be performed modulo 2^16. With this, sequence numbers cycle from 0 to 2^16-1 and start from 0 again when reaching the maximum value. Therefore, special care must be taken with comparisons. For example, the 7 Sequence Numbers below 5 modulo 2^16 are 4,3,2,1,0,65535 and 65534.

Conceptional illustration of In-Window and Out-Of-Range Sequence Numbers for a WINDOW_SIZE of 8 (The proposed WINDOW_SIZE constant is defined in Section 8)

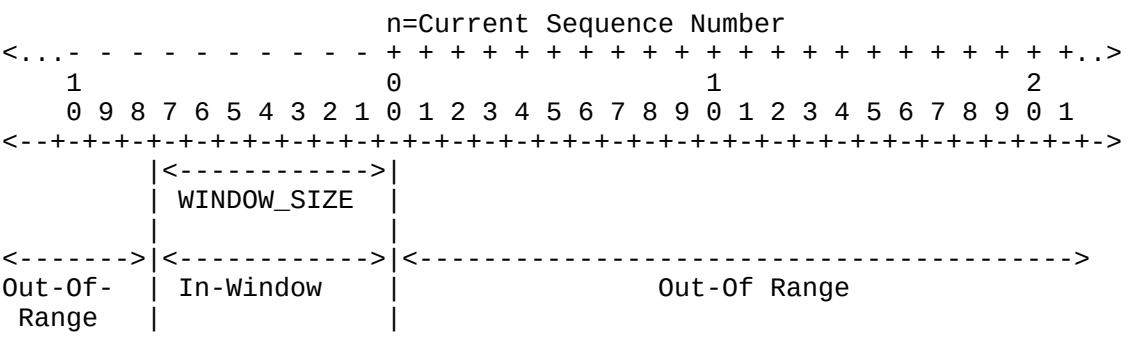

Figure 4

In-Window Sequence Numbers:

The In-Window Sequence Numbers represent the latest sequence numbers. They include the Current Sequence Number from this Originator and the WINDOW_SIZE - 1 Sequence Numbers below it.

The Current Sequence Number of each Originator MUST be maintained, as well as information if an OGM has been received or not for each In-Window Sequence Number.

If an OGM from this Originator with a In-Window Sequence Number is received, the Current Sequence Number will NOT be updated, and therefore the Sliding Window is not moved. It must be memorized that an OGM with Sequence Number has been received.

Out-Of-Range Sequence Numbers:

Out-Of-Range Sequence Numbers are all Sequence Numbers which are not in the In-Window range. They are considered new or next-expected Sequence Numbers.

If an OGM from this Originator with an Out-Of-Range Sequence Number is received, the Current Sequence Number is set to this new Sequence Number. This means that the Sliding Window is moved, and Sequence Numbers which are not in the In-Window Range anymore drop out of the Window. Information if OGMs have been received or not with Sequence Numbers which dropped out of the Window MUST be purged.

5. Flooding Mechanism

The flooding mechanism can be divided into several parts:

- How to generate and broadcast OGMs is described in Section 5.1.
- The reception and evaluation of OGMs is described in Section 5.2.
- The update and usage of the Bidirectional Link Check is explained in Section 5.3.
- The Neighbor Ranking and Best Link detection is described in Section 5.4.
- The rebroadcast mechanism is described in Section 5.5.

5.1. Broadcasting own Originator Messages (OGMs)

Each node MUST periodically generate and broadcast OGMs for each B.A.T.M.A.N. Interface. The Message(s) MUST be broadcasted every ORIGINATOR_INTERVAL via all B.A.T.M.A.N. Interfaces. A jitter may be applied to avoid collisions. The OGM MUST be initialized as follows:

- Version: Set your internal compatibility version.
- Flags: Set the Is-direct-link and Unidirectional flag to zero.
- TTL: Set the TTL to the desired value in the range of TTL_MIN and TTL_MAX.
- Sequence number: On first broadcast set the sequence number to an arbitrary value and increment the field by one for each following broadcast.
- GWFlags: If this host offers an internet connection set the field as described in Section 7.1 otherwise set it to zero.
- GWPort: If this host offers an internet connection set the field to your desired tunneling port otherwise set it to zero.
- Originator Address: Set this field to the IP address of this B.A.T.M.A.N. Interface.

If this Node wants to announce access to non-B.A.T.M.A.N. networks via HNA it SHOULD append an HNA Extension Messages for every network to be announced. See Section 3.3 for a detailed description of that message type.

5.2. Receiving Originator Messages

Upon receiving a general B.A.T.M.A.N. packet a Node MUST perform the following preliminary checks before the packet is further processed:

1. If the OGM contains a version which is different to the own internal version the message MUST be silently dropped (thus, it MUST NOT be further processed or broadcasted).

2. If the sender address of the OGM belongs to one of the B.A.T.M.A.N. interfaces the message MUST be silently dropped as this OGM originated from this Node.

3. If the sender address of the OGM is a broadcast address of an own B.A.T.M.A.N. interface the message MUST be silently dropped.

4. If the Originator Address of the OGM is identical with any of the Nodes' own B.A.T.M.A.N. Interfaces then the OGM has been originated by the Node itself. The processing of the OGM MUST continue as described in Section 5.3 and afterwards silently dropped.

5. If the unidirectional flag of the OGM is set the message MUST be silently dropped.

6. If the OGM has been received via a Bidirectional link AND contains a New Sequence Number (is NOT a duplicate) then the OGM MUST be processed as described in Section 5.4.

7. The OGM has to be rebroadcasted as described in Section 5.5 if:

   * the OGM has been received from a single hop neighbor (sender address equals Originator Address)

   * the OGM was received via a Bidirectional link AND via the Best Link AND is either not a duplicate or has the same TTL as the last packet which was not a duplicate (last TTL)

5.3. Bidirectional Link Check

A Bidirectional Link check is used to verify that a detected link to a given neighbor can be used in both directions.

Therefore the Sequence Number of each self-originated OGM (re-broadcasted by a direct link neighbor) for each Interface must be recorded (Bidirect Link Sequence Number) if:

- the self-originated OGM has been received via the Interface for which the OGM has been generated
- the direct-link-flag is set
- the Sequence Number matches the Sequence Number send with the last OGM broadcasted for that Interface

The bidirectional link check succeeds if the last originated Sequence number does not differ more than BI_LINK_TIMEOUT from the recorded Sequence number.

5.4. Neighbor Ranking

Upon reception of an OGM from another node the following must be performed:

- The Packet Count MUST be updated.
- If the OGMs Sequence Number is newer than the Current Sequence Number:
  - The new Current Sequence Number MUST be set to the Sequence Number contained in the received OGM.
  - The Last TTL of this neighbor MUST be updated.
  - The Sliding Windows of all known links to the Originator of the OGM must be updated (purged) to reflect the new upper and lower boundaries of the Ranking Range. The Sequence Number of the received OGM must be added to the Sliding Window representing the link via which the OGM has been received.
- If the Sliding Window of the link via which the OGM has been received contains the most (In-Ranking-Range) Sequence Numbers then this link is said to be the new Best Link to the Originator of the OGM. Otherwise the previously considered Best Link MUST NOT change.

5.5. Re-broadcasting other nodes' OGMs

When an OGM is to be re-broadcasted some of the message fields MUST be changed others MUST be left unchanged. All fields not mentioned in the following section remain untouched:

- The TTL must be decremented by one. If the TTL becomes zero (after the decrementation) the packet MUST be dropped.

- The Is-direct-link MUST be set if the OGM has been received from a Direct Link Neighbor AND if it is re-broadcasted via the link via which it has been received.

- The Unidirectional flag must be set if an OGM is to be re-broadcasted that has been received via an unidirectional link.

6. Routing

In order to maintain the routing table of a B.A.T.M.A.N. node, the routing daemon keeps track of incoming new OGMs and maintains a list of all Originators which have sent Originator messages as shown in section Section 4.

B.A.T.M.A.N. maintains one dedicated routing entry for each known Originator and HNA announcement. Each routing entry defines the outgoing B.A.T.M.A.N. interface and the IP address of the next-hop direct-link neighbor towards the corresponding Originator. B.A.T.M.A.N. must add a host route to all Originators, even if they are link-local bidirectional single-hop neighbors.

6.1. Route Selection and Establishing

If a OGM from an unknown Originator or to an unknown network/host via HNA is received, it will be added to the routing table, and the best ranking link-local bidirectional neighbor is selected as gateway to the destination. If the destination is a host, a host route will be added via the best ranking bidirectional single-hop neighbor for the destination. If the destination is a network, announced by HNA information included in a OGM message, a network route is added via the best ranking bidirectional single-hop neighbor. A bidirectional single-hop neighbor may or may not be selected as gateway to itself. In case a single-hop Originator is not the best gateway to itself, an host route via another bidirectional single-hop neighbor MUST be chosen.

If the best ranking neighbor to the destination changes, the routing table must be updated.

The gateway of each host route to an Originator must be in sync with the Best Link identified for the Originator, as described in Section Section 5.4. The gateway of each HNA related host or network route must be in sync with the host route of the Originator that ownes the corresponding HNA message.

6.2. Route Deletion

In case a node does not receive a single OGM or HNA from a known Originator for a time longer than the sliding WINDOW_SIZE and the PURGE_TIMEOUT interval, the route is considered to be expired and will be removed from the routing table.

6.3. Opportunistic Routing Deletion Policy

B.A.T.M.A.N. should behave opportunistic when deleting routes: The suggested purging intervals for routes should be long, compared to the sliding window size (Recommended value: PURGE_TIMEOUT = 10 x WINDOW_SIZE x ORIGINATOR_INTERVAL).

6.3.1. Opportunistic Routing Deletion Policy Consideration

A routing entry to a destination that is no longer working, is a minor problem in terms of managing network traffic efficiently. The only disadvantage is, that a node may utilize the network trying to send information to an unreachable destination for a while, before giving up. On the other hand, having no routing entry to a destination that would otherwise be accessible, is problematic in terms of routing functionality. To avoid an overflow of routing information, the routing table is purged from expired entrys according to the PURGE_TIMEOUT interval. However, as soon as new OGMs from a destination are received, the routing entry is updated if a change in the network topology has occurred.

7. Gateway

7.1. Gateway Announcement

A B.A.T.M.A.N. node with access to the internet and routing capabilities MAY act as a internet Gateway. The Gateway is announced with the GWFlags transmitted within the B.A.T.M.A.N.-OGM packets. If the node does not provide access to the internet, it MUST set GWFlags to 0. Otherwise, the GWFlags contains the provided bandwidth encoded as described below. The providing node SHOULD set this value to the best approximate estimate of available bandwidth.

The GWFlags fields.

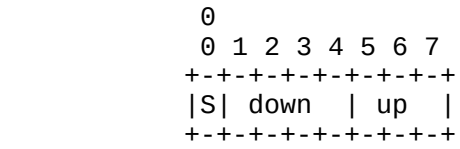

Figure 5

The GWFlags encodes the approximate available bandwidth in kbit per second. The downstream and upstream bandwidth is calculated based on the fields, which are interpreted as binary numbers:

downstream bandwidth = 32 * (S + 2) * 2^down kbit/s

upstream bandwidth = ((up +1) * (downstream bandwidth)) / 8 kbit/s

(annotation: 2^x means 2 raised to the power of x)

7.2. Gateway Selection

The B.A.T.M.A.N.-nodes can determine the gateway in several ways. The individual node on the network can either choose to decide upon the gateway to be used according to the download speed and connection quality or only according to the connection speed with the gateway itself or as a suitable solution for mobile nodes only by checking for the gateway with the best download speed, but this inherits a frequent change in the gateway used.

It is suggested that the B.A.T.M.A.N.-nodes should, in order to guarantee functionality, be able to determine and decide upon their internet-gateways in multiple ways. It would seem useful that the individual user could, for example, be able to choose any given combination of the download speed and the connection strength to the internet-gateway i.e. only looking at a combination of the conditions noted or decide to disregard one. This might be important for mobile nodes for example as it could be their priority to have a good connection to their gateway rather than having the focus on their internet-connection’s speed. On the other hand would this allow for static users to accept a worse connection to the gateway itself to have a faster connection to the internet. And in some cases it might prove useful to combine both methods although a dynamically chosen internet-gateway always brings with it the possibility of all connections being reset due to switching from one gateway to another. Hence it is strongly suggested that the routing-protocol should include the possibility for the user to set his gateway statically

and not having the protocol deciding upon the best route to the internet but using this as a fallback method should the gateway configured by the user not be reachable.

7.3. Gateway Tunneling/Encapsulation

A GW-client node tunnels all data to the internet (all IP packets with a destination address that does only match the default route) via a selected GW node. No encapsulation is used for packets from the internet to the GW-client nodes. The GW-client node encapsulates the internet data into a IP/UDP datagram and forwards the encapsulated data to the selected GW node. The GW node identifies the encapsulated packets based on the port number of the outer UDP header. It decapsulates the original packet and forwards it to its' original destination. This procedure is completely stateless.

For encapsulation, a GW-client node MUST set the outer IP header source and destination address to the Originator Address of the GW-Client node and the GW node. The outer UDP source and destination MUST be set to the GW Port number given by the OGM of the GW node. The inner IP header and all following data represents the original IP packet. All data of the inner IP packet MUST be left unchanged. If the size of the original IP packet does not fit into the payload section of the outer UDP datagram the packet must be dropped. If virtual interfaces are used to integrate an implementation of the B.A.T.M.A.N protocol into a network environment then the maximum transfer unit (MTU) of the virtual interface should reflect the maximum payload size of the inner UDP datagram.

7.4. Gateway hopping (testing/accepting)

test the gateway (is it connected to the internet?) choose a better gateway if its not available.

8. Proposed Values for Constants

VERSION = 4

TTL_MIN = 2

TTL_MAX = 255

SEQNO_MAX = 65535

BROADCAST_DELAY_MAX (Milliseconds) = 100

ORIGINATOR_INTERVAL (Milliseconds) = 1000

ORIGINATOR_INTERVAL_JITTER (Milliseconds)= 200

WINDOW_SIZE = 128

PURGE_TIMEOUT = 10 x WINDOW_SIZE x ORIGINATOR_INTERVAL

9. IANA Considerations

This memo includes no request to IANA.

All drafts are required to have an IANA considerations section (see the update of RFC 2434 [I-D.narten-iana-considerations-rfc2434bis] for a guide). If the draft does not require IANA to do anything, the section contains an explicit statement that this is the case (as above). If there are no requirements for IANA, the section will be removed during conversion into an RFC by the RFC Editor.

10. Security Considerations

Routing protocols have to rely on information from other nodes in the network. Therefore they are susceptible to various attacks and B.A.T.M.A.N. being one of these protocols has to bear with them. The B.A.T.M.A.N. protocol can be enhanced by the use of common encryption and authentication technologies to insure that routing information is only accepted from trusted nodes. To increase the level of security, all information on the wireless layer itself may also be encrypted. However, these approaches do not solve the challenges of a mesh network consisting of non-authenticated, non-trusted peers and are not in the scope of this document. In case there is no closed trusted group of peers, the routing algorithm itself has to be robust against false protocol informations. B.A.T.M.A.N.'s protocol design inherently limits the impact of different attack vectors.

10.1. Confidentiality

A B.A.T.M.A.N. Node knows of the existence of all other nodes in the network which are in the range of multi-hop communication links, but due to its design it does not know the whole topology of the network. A Nodes' topology view is limited to a one hop horizon. B.A.T.M.A.N. accepts packets from arbitrary sources and builds its routing table by analyzing the statistics of received Originator Messages.

10.2. Overflow of routing entries

A malicious host could send Originator Messages that are announcing the existence of non-existing nodes to cause an overflow of routing

entrys or excessive cpu load and memory consumption. This attack can be intercepted by sanity checks. If the number of routing entries goes through the roof, Originators with very low Originator Message count must be removed.

10.3. Route manipulation

An attacker can also generate OGMs from an existing Originator with continuing valid Sequence Numbers that he actually didn't receive - in order to manipulate other hosts routing, and redirect the route to the destination to itself. Since routing decisions are based on statistical analysis of the number of incoming Originator Messages, rather than on information contained in packets, the attacker has to generate many falsified protocol messages. Like valid protocol messages, phony messages created by the attacker are subject to packet loss. If an attacker wants to make sure that a route via his controlled host will be chosen, he has to win the ranking towards the destination continuously. This limits the range of successful attacks to areas where the attacker can deliver enough false messages to override valid messages.

If the Sequence numbers sent by the attacker differ more than the sliding window size, the victims will assume that the other host has restarted and will purge the ranking. The attacker can constantly generate OGMs with Sequence numbers that induce all receiving nodes to purge the ranking every time they receive a phony OGM. But every time a valid OGM is received by the victims, his phony routing information will be purged again. This limits the range and duration of a successful attack.

The attacker may send phony OGMs for an existing Originator, that are a few counts ahead of the real Sequence Number. This way the packets from the attacker will be preferred in the ranking, and will not induce the victims to purge the ranking. However if the number of phony OGMs delivered to the victim is too low to win the ranking, the attack will have no effect. Again, the range of an attack is limited.

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, Current Status BEST CURRENT PRACTICE, March 1997.

11.2. Informative References

[I-D.narten-iana-considerations-rfc2434bis]
Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", draft-narten-iana-considerations-rfc2434bis-09 (work in progress), March 2008.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

Appendix A. Contributors

This specification is the result of the joint efforts of the following contributors -- listed alphabetically.

Andreas Langer

Axel Neumann

Corinna (Elektra) Aichele

Felix Fietkau

Ludger Schmudde

Marek Lindner

Simon Wunderlich

Thomas Lopatic

Authors' Addresses

Axel Neumann

Email: axel@open-mesh.net

Corinna (Elektra) Aichele

Email: elektra@open-mesh.net

Marek Lindner

Email: lindner_marek@yahoo.de

Simon Wunderlich

Email: siwu@hrz.tu-chemnitz.de

Full Copyright Statement

Intellectual Property

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).