Messages for the Masses

State of Multicast in batman-adv and Gluon

Linus Lüssing (T_X)

Wireless Battlemesh v16, Cyprus

May, 2024

What is batman-adv

What is batman-adv

- A layer 2 mesh routing protocol

What is batman-adv

- A layer 2 mesh routing protocol
- Encapsulates ethernet frames

What is batman-adv

- A layer 2 mesh routing protocol
- Encapsulates ethernet frames
- Network protocol agnostic

What is batman-adv

- A layer 2 mesh routing protocol
- Encapsulates ethernet frames
- Network protocol agnostic
- Supports/requires multicast

What is Gluon

What is Gluon

- A firmware framework for public mesh networks

What is Gluon

- A firmware framework for public mesh networks
- Based on OpenWrt

What is Gluon

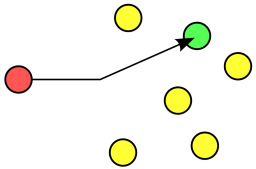
- A firmware framework for public mesh networks
- Based on OpenWrt
- Integrates: batman-adv

What is Gluon

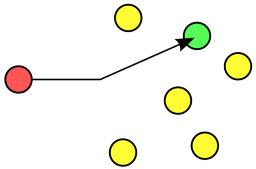
- A firmware framework for public mesh networks
- Based on OpenWrt
- Integrates: batman-adv
- Popular in Freifunk communities

What is Multicast

What is Multicast

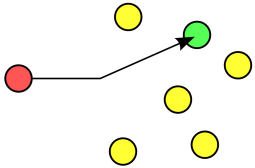


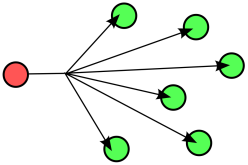
What is Multicast



- Unicast

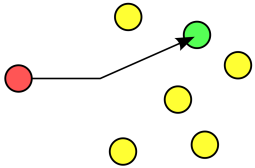
What is Multicast



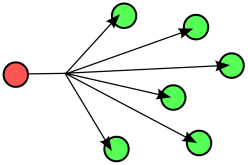


- Unicast

What is Multicast

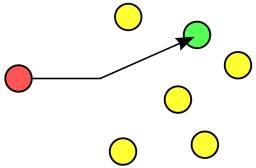


- Unicast

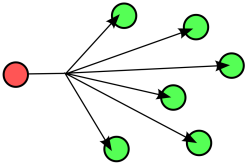


- Broadcast

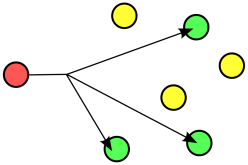
What is Multicast



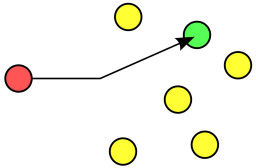
- Unicast



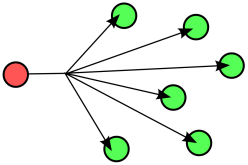
- Broadcast



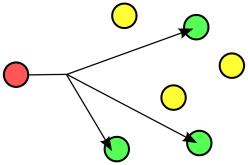
What is Multicast



- Unicast



- Broadcast



- Multicast

Motivation for (proper) Multicast

Motivation for (proper) Multicast

- less layer 2 overhead with batman-adv

Motivation for (proper) Multicast

- less layer 2 overhead with batman-adv
- more throughput (i.e. IPTV / media streaming)

Motivation for (proper) Multicast

- less layer 2 overhead with batman-adv
- more throughput (i.e. IPTV / media streaming)
- more decentralization, vs. server $\rightarrow$ client model

Motivation for (proper) Multicast

- less layer 2 overhead with batman-adv
- more throughput (i.e. IPTV / media streaming)
- more decentralization, vs. server → client model
  - be your own radio/streaming station

Motivation for (proper) Multicast

- less layer 2 overhead with batman-adv
- more throughput (i.e. IPTV / media streaming)
- more decentralization, vs. server → client model
  - be your own radio/streaming station
  - decentral service discovery (mDNS, SAP, ...)

Classic flooding (v0.1)

Classic flooding (v0.1)

- Each neighbor rebroadcasts

Classic flooding (v0.1)

- Each neighbor rebroadcasts
- Loops?

Classic flooding (v0.1)

- Each neighbor rebroadcasts
- Loops?
- $\Rightarrow$ Sequence numbers!

Classic flooding (v0.1)

- Each neighbor rebroadcasts
- Loops?
- $\Rightarrow$ Sequence numbers!
- Since v0.2 (2009): 3x rebroadcasts

Broadcast Avoidances (v2016.0)

Broadcast Avoidances (v2016.0)

- Don't (re)broadcast on an interface if:

Broadcast Avoidances (v2016.0)

- Don't (re)broadcast on an interface if:
  - No neighbor node

Broadcast Avoidances (v2016.0)

- Don't (re)broadcast on an interface if:
  - No neighbor node
  - Neighbor is previous transmitter

Broadcast Avoidances (v2016.0)

- Don't (re)broadcast on an interface if:
  - No neighbor node
  - Neighbor is previous transmitter
  - Neighbor is originator

Broadcast Avoidances (v2016.0)

- Don't (re)broadcast on an interface if:
  - No neighbor node
  - Neighbor is previous transmitter
  - Neighbor is originator
- Also applied to: OGMv2/BATMAN V

Gateway Feature (v2011.0.0)

Gateway Feature (v2011.0.0)

- Dynamic Host Configuration Protocol (DHCP):

Gateway Feature (v2011.0.0)

- Dynamic Host Configuration Protocol (DHCP):
  - IP + routes → host assignment

Gateway Feature (v2011.0.0)

- Dynamic Host Configuration Protocol (DHCP):
  - IP + routes $\rightarrow$ host assignment
- Gateway Feature:

Gateway Feature (v2011.0.0)

- Dynamic Host Configuration Protocol (DHCP):
  - IP + routes → host assignment
- Gateway Feature:
  - DHCP servers announced through batman-adv

Gateway Feature (v2011.0.0)

- Dynamic Host Configuration Protocol (DHCP):
  - IP + routes $\rightarrow$ host assignment
- Gateway Feature:
  - DHCP servers announced through batman-adv
  - DHCP from client: to "best" gateway via unicast

Distributed ARP Table (v2013.0.0)

Distributed ARP Table (v2013.0.0)

- Address Resolution Protocol: IPv4 $\rightarrow$ MAC

Distributed ARP Table (v2013.0.0)

- Address Resolution Protocol: IPv4 $\rightarrow$ MAC
- DAT: Distributed Hash Table (DHT) for ARP

Distributed ARP Table (v2013.0.0)

- Address Resolution Protocol: IPv4 $\rightarrow$ MAC
- DAT: Distributed Hash Table (DHT) for ARP
  - ARP Reply: (IPv4, MAC) to 3x nodes

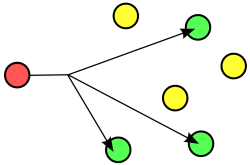
Distributed ARP Table (v2013.0.0)

- Address Resolution Protocol: IPv4 $\rightarrow$ MAC
- DAT: Distributed Hash Table (DHT) for ARP
  - ARP Reply: (IPv4, MAC) to 3x nodes
  - ARP Request: (IPv4, MAC) from these 3 nodes

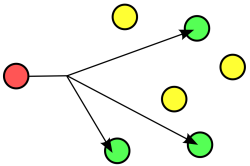
Distributed ARP Table (v2013.0.0)

- Address Resolution Protocol: IPv4 $\rightarrow$ MAC
- DAT: Distributed Hash Table (DHT) for ARP
  - ARP Reply: (IPv4, MAC) to 3x nodes
  - ARP Request: (IPv4, MAC) from these 3 nodes
  - $\Rightarrow$ via unicast, no broadcasting

Generic Multicast Optimizations: Learning

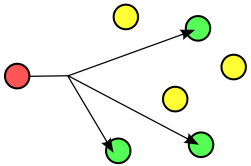


Generic Multicast Optimizations: Learning



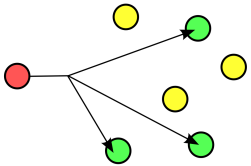
- batman-adv learns IP multicast listeners + multicast routers

Generic Multicast Optimizations: Learning



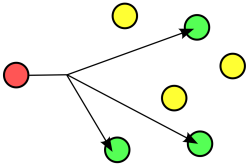
- batman-adv learns IP multicast listeners + multicast routers
  - from local interfaces / kernel:

Generic Multicast Optimizations: Learning



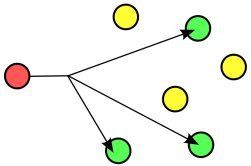
- batman-adv learns IP multicast listeners + multicast routers
  - from local interfaces / kernel:
    - Listeners: $ ip maddr show

Generic Multicast Optimizations: Learning



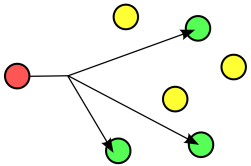
- batman-adv learns IP multicast listeners + multicast routers
  - from local interfaces / kernel:
    - Listeners: $ ip maddr show
    - Routers: $ cat /proc/sys/net/ipv{4,6}/conf/<iface>/mc_forwarding

Generic Multicast Optimizations: Learning



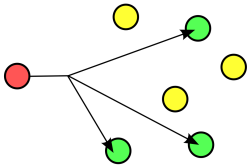
- batman-adv learns IP multicast listeners + multicast routers
  - from local interfaces / kernel:
    - Listeners: $ ip maddr show
    - Routers: $ cat /proc/sys/net/ipv{4,6}/conf/<iface>/mc_forwarding
  - via IGMP (IPv4) / MLD (IPv6) snooping (through the Linux bridge)

Generic Multicast Optimizations: Learning



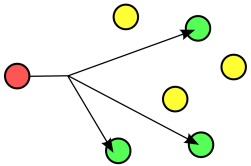
- batman-adv learns IP multicast listeners + multicast routers
  - from local interfaces / kernel:
    - Listeners: $ ip maddr show
    - Routers: $ cat /proc/sys/net/ipv{4,6}/conf/<iface>/mc_forwarding
  - via IGMP (IPv4) / MLD (IPv6) snooping (through the Linux bridge)
  - via MRD snooping (through the Linux bridge)

Generic Multicast Optimizations: Learning



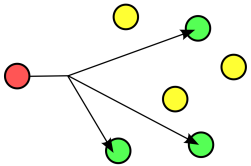
- batman-adv learns IP multicast listeners + multicast routers
  - from local interfaces / kernel:
    - Listeners: $ ip maddr show
    - Routers: $ cat /proc/sys/net/ipv{4,6}/conf/<iface>/mc_forwarding
  - via IGMP (IPv4) / MLD (IPv6) snooping (through the Linux bridge)
  - via MRD snooping (through the Linux bridge)
- Teaching:

Generic Multicast Optimizations: Learning



- batman-adv learns IP multicast listeners + multicast routers
  - from local interfaces / kernel:
    - Listeners: $ ip maddr show
    - Routers: $ cat /proc/sys/net/ipv{4,6}/conf/<iface>/mc_forwarding
  - via IGMP (IPv4) / MLD (IPv6) snooping (through the Linux bridge)
  - via MRD snooping (through the Linux bridge)
- Teaching:
  - via batman-adv translation table (TT)

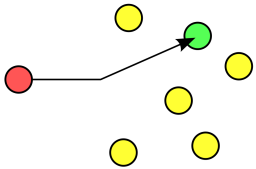
Generic Multicast Optimizations: Learning



- batman-adv learns IP multicast listeners + multicast routers
  - from local interfaces / kernel:
    - Listeners: $ ip maddr show
    - Routers: $ cat /proc/sys/net/ipv{4,6}/conf/<iface>/mc_forwarding
  - via IGMP (IPv4) / MLD (IPv6) snooping (through the Linux bridge)
  - via MRD snooping (through the Linux bridge)
- Teaching:
  - via batman-adv translation table (TT)
  - via batman-adv multicast TVLV on OGMs

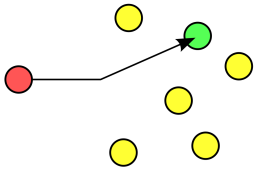
Generic Multicast Optimizations: ICMPv6 ND

Generic Multicast Optimizations: ICMPv6 ND



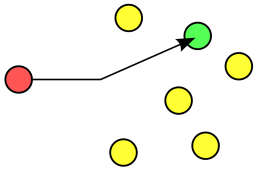
- Less ICMPv6 Neighbor Discovery overhead

Generic Multicast Optimizations: ICMPv6 ND



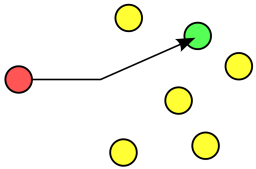
- Less ICMPv6 Neighbor Discovery overhead
- ICMPv6 Neighbor Solicitation for:

Generic Multicast Optimizations: ICMPv6 ND



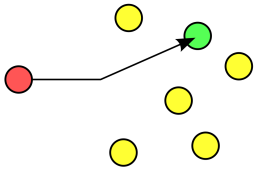
- Less ICMPv6 Neighbor Discovery overhead
- ICMPv6 Neighbor Solicitation for:
  - Duplicate Address Detection: typ. 0 listeners

Generic Multicast Optimizations: ICMPv6 ND



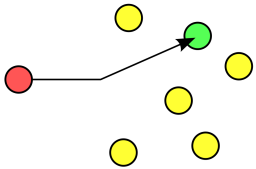
- Less ICMPv6 Neighbor Discovery overhead
- ICMPv6 Neighbor Solicitation for:
  - Duplicate Address Detection: typ. 0 listeners
  - Address Resolution (IPv6 $\rightarrow$ MAC): typ. 1 listener

Generic Multicast Optimizations: ICMPv6 ND



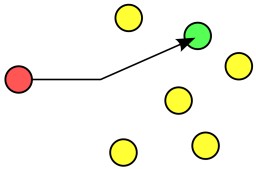
- Less ICMPv6 Neighbor Discovery overhead
- ICMPv6 Neighbor Solicitation for:
  - Duplicate Address Detection: typ. 0 listeners
  - Address Resolution (IPv6 $\rightarrow$ MAC): typ. 1 listener
- Solicited-node multicast address:

Generic Multicast Optimizations: ICMPv6 ND



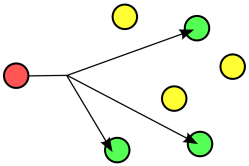
- Less ICMPv6 Neighbor Discovery overhead
- ICMPv6 Neighbor Solicitation for:
  - Duplicate Address Detection: typ. 0 listeners
  - Address Resolution (IPv6 $\rightarrow$ MAC): typ. 1 listener
- Solicited-node multicast address:
  - For each IPv6 unicast address: (typ.) 1x multicast address

Generic Multicast Optimizations: ICMPv6 ND

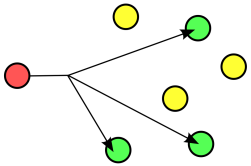


- Less ICMPv6 Neighbor Discovery overhead
- ICMPv6 Neighbor Solicitation for:
  - Duplicate Address Detection: typ. 0 listeners
  - Address Resolution (IPv6 $\rightarrow$ MAC): typ. 1 listener
- Solicited-node multicast address:
  - For each IPv6 unicast address: (typ.) 1x multicast address
  - Example: fe80::cc97:a1ff:fe7b:451 $\rightarrow$ ff02::1:ff7b:451

Generic Multicast Optimizations: Multicast Packet Type (v2024.0, NEW)

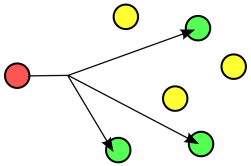


Generic Multicast Optimizations: Multicast Packet Type (v2024.0, NEW)



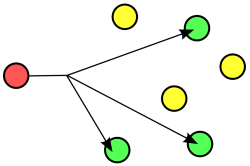
- new batman-adv multicast packet type:

Generic Multicast Optimizations: Multicast Packet Type (v2024.0, NEW)



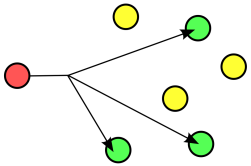
- new batman-adv multicast packet type:
  - 1 packet, n destination node MAC addresses

Generic Multicast Optimizations: Multicast Packet Type (v2024.0, NEW)



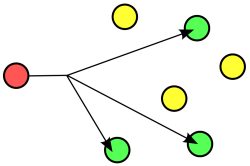
- new batman-adv multicast packet type:
  - 1 packet, n destination node MAC addresses
- hop-by-hop forwarding via unicast

Generic Multicast Optimizations: Multicast Packet Type (v2024.0, NEW)



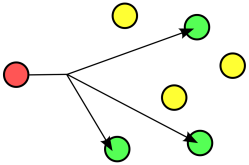
- new batman-adv multicast packet type:
  - 1 packet, n destination node MAC addresses
- hop-by-hop forwarding via unicast
- each forwarder: (pot.) splits packet

Generic Multicast Optimizations: Multicast Packet Type (v2024.0, NEW)

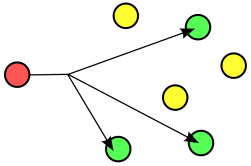


- new batman-adv multicast packet type:
  - 1 packet, n destination node MAC addresses
- hop-by-hop forwarding via unicast
- each forwarder: (pot.) splits packet
- $\Rightarrow$ reduced overhead / more throughput

Generic Multicast Optimizations: Limitations

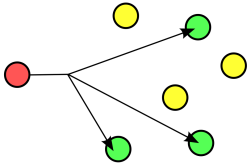


Generic Multicast Optimizations: Limitations



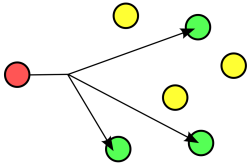
- all nodes need support

Generic Multicast Optimizations: Limitations



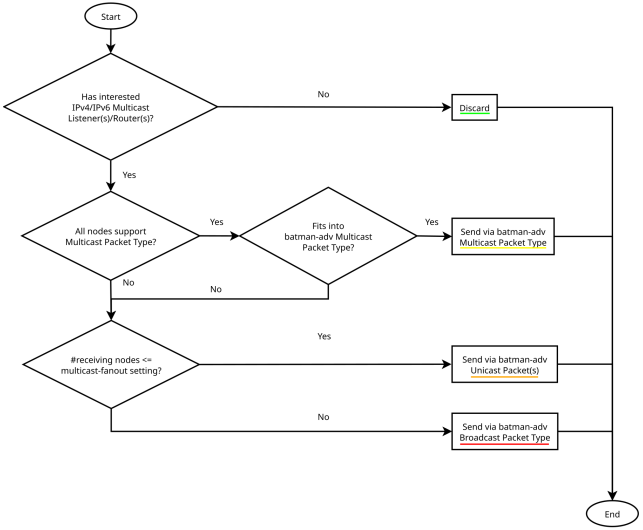
- all nodes need support
- all nodes need interface MTUs $>= 1280$
  $\Rightarrow$ no MTU path discovery, no fragmentation support

Generic Multicast Optimizations: Limitations



- all nodes need support
- all nodes need interface MTUs >= 1280
  $\Rightarrow$ no MTU path discovery, no fragmentation support
- maximum (depending on payload size):
  2 to 196 destination nodes

Generic Multicast Optimizations: Fallbacks



Gluon v2023.2(.x)

Gluon v2023.2(.x)

- bridge MLD snooping enabled
  $\Rightarrow$ AP$\rightarrow$client, AP$\rightarrow$mesh

Gluon v2023.2(.x)

- bridge MLD snooping enabled
  $\Rightarrow$ AP$\rightarrow$client, AP$\rightarrow$mesh
- bridge MLD querier enabled,
  interval: 20s, response interval: 5s, robustness variable: 9

Gluon v2023.2(.x)

- bridge MLD snooping enabled
  $\Rightarrow$ AP$\rightarrow$client, AP$\rightarrow$mesh
- bridge MLD querier enabled,
  interval: 20s, response interval: 5s, robustness variable: 9
- batman-adv multicast optimizations enabled
  $\Rightarrow$ node$\rightarrow$node $\Rightarrow$ forwarding (pot.) via multiple unicast packets
  (not yet: batman-adv multicast packet type)

Gluon v2023.2(.x)

- bridge MLD snooping enabled
  $\Rightarrow$ AP$\rightarrow$client, AP$\rightarrow$mesh
- bridge MLD querier enabled,
  interval: 20s, response interval: 5s, robustness variable: 9
- batman-adv multicast optimizations enabled
  $\Rightarrow$ node$\rightarrow$node $\Rightarrow$ forwarding (pot.) via multiple unicast packets
  (not yet: batman-adv multicast packet type)
- now with MRD support, to detect multicast routers
  $\Rightarrow$ thanks to OpenWrt 23.05 (batman-adv v2023.1, Linux bridge v5.15) $\Rightarrow$ IPv6 routeable multicast address support
  (ffXY::..., with Y $>$ 2)

OpenWrt Wifi multicast-to-unicast

OpenWrt Wifi multicast-to-unicast

- OpenWrt WiFi multicast-to-unicast enabled

OpenWrt Wifi multicast-to-unicast

- OpenWrt WiFi multicast-to-unicast enabled
  - Wifi isolation setting:
    $\Rightarrow$ all multicast to bridge

OpenWrt Wifi multicast-to-unicast

- OpenWrt WiFi multicast-to-unicast enabled
  - Wifi isolation setting:
    $\Rightarrow$ all multicast to bridge
  - bridge hair-pin mode:
    $\Rightarrow$ after bridge MLD snooping, reflect back

OpenWrt Wifi multicast-to-unicast

- OpenWrt WiFi multicast-to-unicast enabled
  - Wifi isolation setting:
    $\Rightarrow$ all multicast to bridge
  - bridge hair-pin mode:
    $\Rightarrow$ after bridge MLD snooping, reflect back
  - bridge multicast-to-unicast,
    for each: AP$\rightarrow$STA/client

OpenWrt Wifi multicast-to-unicast

- OpenWrt WiFi multicast-to-unicast enabled
  - Wifi isolation setting:
    $\Rightarrow$ all multicast to bridge
  - bridge hair-pin mode:
    $\Rightarrow$ after bridge MLD snooping, reflect back
  - bridge multicast-to-unicast,
    for each: AP$\rightarrow$STA/client
    - copy multicast frame

OpenWrt Wifi multicast-to-unicast

- OpenWrt WiFi multicast-to-unicast enabled
  - Wifi isolation setting:
    $\Rightarrow$ all multicast to bridge
  - bridge hair-pin mode:
    $\Rightarrow$ after bridge MLD snooping, reflect back
  - bridge multicast-to-unicast,
    for each: AP$\rightarrow$STA/client
    - copy multicast frame
    - set ethernet destination to unicast address of STA

OpenWrt Wifi multicast-to-unicast

- OpenWrt WiFi multicast-to-unicast enabled
  - Wifi isolation setting:
    $\Rightarrow$ all multicast to bridge
  - bridge hair-pin mode:
    $\Rightarrow$ after bridge MLD snooping, reflect back
  - bridge multicast-to-unicast,
    for each: AP$\rightarrow$STA/client
    - copy multicast frame
    - set ethernet destination to unicast address of STA
- (typ.) higher Wifi rate

OpenWrt Wifi multicast-to-unicast

- OpenWrt WiFi multicast-to-unicast enabled
  - Wifi isolation setting:
    $\Rightarrow$ all multicast to bridge
  - bridge hair-pin mode:
    $\Rightarrow$ after bridge MLD snooping, reflect back
  - bridge multicast-to-unicast,
    for each: AP$\rightarrow$STA/client
    - copy multicast frame
    - set ethernet destination to unicast address of STA
- (typ.) higher Wifi rate
- (and if not: "new" airtime fairness Linux WLAN patches)

OpenWrt Wifi multicast-to-unicast

- OpenWrt WiFi multicast-to-unicast enabled
  - Wifi isolation setting:
    $\Rightarrow$ all multicast to bridge
  - bridge hair-pin mode:
    $\Rightarrow$ after bridge MLD snooping, reflect back
  - bridge multicast-to-unicast,
    for each: AP$\rightarrow$STA/client
    - copy multicast frame
    - set ethernet destination to unicast address of STA
- (typ.) higher Wifi rate
- (and if not: "new" airtime fairness Linux WLAN patches)
- ACK'd + retried transmissions $\Rightarrow$ reliability

Firewall: gluon-ebtables-filter-multicast package

Firewall: gluon-ebtables-filter-multicast package

- Filter all batman-adv broadcast packets, avoid classic flooding! (default)

Firewall: gluon-ebtables-filter-multicast package

- Filter all batman-adv broadcast packets,
  avoid classic flooding! (default)
- no-flood mark batman-adv patch (not yet upstream)

Firewall: gluon-ebtables-filter-multicast package

- Filter all batman-adv broadcast packets,
  avoid classic flooding! (default)
- no-flood mark batman-adv patch (not yet upstream)
- Unless:

Firewall: gluon-ebtables-filter-multicast package

- Filter all batman-adv broadcast packets,
  avoid classic flooding! (default)
- no-flood mark batman-adv patch (not yet upstream)
- Unless:
  - ARP

Firewall: gluon-ebtables-filter-multicast package

- Filter all batman-adv broadcast packets,
  avoid classic flooding! (default)
- no-flood mark batman-adv patch (not yet upstream)
- Unless:
  - ARP
  - DHCP

Firewall: gluon-ebtables-filter-multicast package

- Filter all batman-adv broadcast packets,
  avoid classic flooding! (default)
- no-flood mark batman-adv patch (not yet upstream)
- Unless:
  - ARP
  - DHCP
  - IGMP

Firewall: gluon-ebtables-filter-multicast package

- Filter all batman-adv broadcast packets,
  avoid classic flooding! (default)
- no-flood mark batman-adv patch (not yet upstream)
- Unless:
  - ARP
  - DHCP
  - IGMP
  - ICMPv6, except:

Firewall: gluon-ebtables-filter-multicast package

- Filter all batman-adv broadcast packets,
  avoid classic flooding! (default)
- no-flood mark batman-adv patch (not yet upstream)
- Unless:
  - ARP
  - DHCP
  - IGMP
  - ICMPv6, except:
    - Echo Requests

Firewall: gluon-ebtables-filter-multicast package

- Filter all batman-adv broadcast packets,
  avoid classic flooding! (default)
- no-flood mark batman-adv patch (not yet upstream)
- Unless:
  - ARP
  - DHCP
  - IGMP
  - ICMPv6, except:
    - Echo Requests
    - Node Information Queries

Firewall: gluon-ebtables-filter-multicast package

- Filter all batman-adv broadcast packets, avoid classic flooding! (default)
- no-flood mark batman-adv patch (not yet upstream)
- Unless:
  - ARP
  - DHCP
  - IGMP
  - ICMPv6, except:
    - Echo Requests
    - Node Information Queries
    - MLD queries

Firewall: gluon-ebtables-filter-multicast package

- Filter all batman-adv broadcast packets, avoid classic flooding! (default)
- no-flood mark batman-adv patch (not yet upstream)
- Unless:
  - ARP
  - DHCP
  - IGMP
  - ICMPv6, except:
    - Echo Requests
    - Node Information Queries
    - MLD queries
  - ICMPv6 MLD report, if allowed in config

Firewall: gluon-ebtables-filter-multicast package

- Filter all batman-adv broadcast packets, avoid classic flooding! (default)
- no-flood mark batman-adv patch (not yet upstream)
- Unless:
  - ARP
  - DHCP
  - IGMP
  - ICMPv6, except:
    - Echo Requests
    - Node Information Queries
    - MLD queries
  - ICMPv6 MLD report, if allowed in config
  - Bittorrent Local Peer Discovery

Firewall: gluon-ebtables-filter-multicast package

- Filter all batman-adv broadcast packets, avoid classic flooding! (default)
- no-flood mark batman-adv patch (not yet upstream)
- Unless:
  - ARP
  - DHCP
  - IGMP
  - ICMPv6, except:
    - Echo Requests
    - Node Information Queries
    - MLD queries
  - ICMPv6 MLD report, if allowed in config
  - Bittorrent Local Peer Discovery
  - Babel, OSPF, RIPng

Firewall: gluon-ebtables-limit-arp

- issues with IP range/port scanning apps
- solution: rate-limit ARP
- unless: in batman-adv DAT cache

Gluon wireless settings

Gluon wireless settings

- increased/configurable multicast rate on 11s

Gluon wireless settings

- increased/configurable multicast rate on 11s
- (typ.) set to 12-24 MBit/s by community

Gluon wireless settings

- increased/configurable multicast rate on 11s
- (typ.) set to 12-24 MBit/s by community
- better route decision for BATMAN IV

Gluon wireless settings

- increased/configurable multicast rate on 11s
- (typ.) set to 12-24 MBit/s by community
- better route decision for BATMAN IV
- less multicast/OGM overhead (traded for less reach)

Changes at Freifunk Lübeck

Changes at Freifunk Lübeck

- updated batman-adv to v2024.0:
  $\Rightarrow$ batman-adv multicast packet type support

Changes at Freifunk Lübeck

- updated batman-adv to v2024.0:
  $\Rightarrow$ batman-adv multicast packet type support
- (gluon-ebtables-)brmldproxy package

Changes at Freifunk Lübeck

- updated batman-adv to v2024.0:
  $\Rightarrow$ batman-adv multicast packet type support
- (gluon-ebtables-)brmldproxy package
- Layer 3 routing with Freifunk Vogtland (pim6sd)

Changes at Freifunk Lübeck: brmldproxy

Changes at Freifunk Lübeck: brmldproxy

- aggregates MLD reports from local clients to mesh

Changes at Freifunk Lübeck: brmldproxy

- aggregates MLD reports from local clients to mesh
- only forwards MLD reports for routeable IPv6 multicast

Changes at Freifunk Lübeck: brmldproxy

- aggregates MLD reports from local clients to mesh
- only forwards MLD reports for routeable IPv6 multicast
- forwards MLD reports only to IPv6 multicast routers
  (batman-adv patch, submission TODO, uses new batman-adv multicast packet type)

Changes at Freifunk Lübeck: brmldproxy

- aggregates MLD reports from local clients to mesh
- only forwards MLD reports for routeable IPv6 multicast
- forwards MLD reports only to IPv6 multicast routers
  (batman-adv patch, submission TODO, uses new batman-adv multicast packet type)
  - no multicast router $\Rightarrow$ no MLD reports to mesh

Changes at Freifunk Lübeck: brmldproxy

- aggregates MLD reports from local clients to mesh
- only forwards MLD reports for routeable IPv6 multicast
- forwards MLD reports only to IPv6 multicast routers
  (batman-adv patch, submission TODO, uses new batman-adv multicast packet type)
  - no multicast router ⇒ no MLD reports to mesh
  - no routeable multicast listener ⇒ no MLD reports to mesh

Changes at Freifunk Lübeck: brmldproxy

- aggregates MLD reports from local clients to mesh
- only forwards MLD reports for routeable IPv6 multicast
- forwards MLD reports only to IPv6 multicast routers
  (batman-adv patch, submission TODO, uses new batman-adv multicast packet type)
  - no multicast router ⇒ no MLD reports to mesh
  - no routeable multicast listener ⇒ no MLD reports to mesh
- ⇒ allows layer 3 multicast routing again

Changes at Freifunk Lübeck: brmldproxy

- aggregates MLD reports from local clients to mesh
- only forwards MLD reports for routeable IPv6 multicast
- forwards MLD reports only to IPv6 multicast routers (batman-adv patch, submission TODO, uses new batman-adv multicast packet type)
  - no multicast router ⇒ no MLD reports to mesh
  - no routeable multicast listener ⇒ no MLD reports to mesh
- ⇒ allows layer 3 multicast routing again
- ⇒ pim6sd between Freifunk Lübeck and Freifunk Vogtland

Changes at Freifunk Lübeck: brmldproxy

- aggregates MLD reports from local clients to mesh
- only forwards MLD reports for routeable IPv6 multicast
- forwards MLD reports only to IPv6 multicast routers (batman-adv patch, submission TODO, uses new batman-adv multicast packet type)
  - no multicast router ⇒ no MLD reports to mesh
  - no routeable multicast listener ⇒ no MLD reports to mesh
- ⇒ allows layer 3 multicast routing again
- ⇒ pim6sd between Freifunk Lübeck and Freifunk Vogtland
- (gluon-ebtables-)brmldproxy PR pending at Gluon

Demo

- mDNS (avahi-discover)
- gstreamer/VLC:
  - RTP multicast audio
  - Freifunk Vogtland ⇔ Freifunk Lübeck: PIM, MRD
  - SAP (Session Announcement Protocol)

- VLC 4 nightly: https://nightlies.videolan.org/
- $ ip -6 route add ff0e::/16 dev wlp1s0 table local
- $ ip -6 route add ff7e::/16 dev wlp1s0 table local

Thx, Questions?

- *https://github.com/freifunk-gluon/gluon*
- *https://www.open-mesh.org/*
- Matrix: #gluon:hackint.org, #batadv:hackint.org
- IRC: #gluon / #batadv on hackint.org
- Mailinglist: gluon@luebeck.freifunk.net, b.a.t.m.a.n@lists.open-mesh.org

License:  – CC-BY-SA-4.0, unless noted otherwise