OpenWrt integration with bird2 (UCI and LuCI) for BGP and Babel protocols

2025-02-02

Pedro Vílchez-Blanco

Abstract

The project's proposal consists on developing OpenWrt's integration with bird2 to support the BGP and Babel routing protocols via UCI and LuCI in a way that is useful for community networks. As a student, I aim to build upon previous work on bird1, which was deprecated in 2023, aiming to modernize and maintain interoperability for community networks. The plan outlines phased milestones to implement, test, and document bird2's integration, ensuring robust support for both routing protocols while preparing for future transitions, including bird3.

Contents

1 Context

Years ago, Eloi Casó successfully participated in the 2014 edition of Google Summer of Code[1] to improve integration of bird with OpenWrt. The code was retired on 2023 because bird1 reached end of life, see commit fa136b7[2].

Bird version 2 comes also with Babel mesh routing protocol support. The Babel Routing Protocol (RFC6126) is a loop-avoiding distance-vector routing protocol that is robust and efficient both in ordinary wired networks and in wireless mesh networks.

The overall idea of this project is to:

1. Upgrade from bird v1 to v2
2. Do some maintenance on existing code
3. Include babel as a protocol and make it interact with bgp both via UCI and LUCI

1.1 Why is relevant to maintain and upgrade OpenWrt bird bindings?

Because it is appreciated by the community, for example, the following internship report, used the `bird1-openwrt` package to enable community networks interoperability through BGP[3]. In fact, in guifi.net we have some nodes that use the `bird1-openwrt` package to make interoperability between the backbone network in BGP and the mesh BMX6 networks [4]

Table 1: Number of mesh nodes in Barcelona per sub-zone as of 2025-02-01

nodes	zone
23	Sant Andreu[5]
41	Sants[6]
12	Raval[7]
14	Vallcarca[8]

[1] https://www.google-melange.com/archive/gsoc/2014/orgs/freifunk/projects/eloicaso.html

[2] https://github.com/openwrt/routing/commit/fa136b70dfb29595083353f168d3b6c4df1d8e6e

[3] https://gitlab.com/g10h4ck/lime-guifi-interop-internship-report/-/blob/master/main.asciidoc

[4] here are some of the bmx6 networks that use the bgp integration:

[5] http://dsg.ac.upc.edu/qmpsa/index.php

[6] http://dsg.ac.upc.edu/qmpsu/index.php

[7] http://dsg.ac.upc.edu/qmprv/index.php

[8] http://dsg.ac.upc.edu/qmpvc/index.php

1.2 Why is relevant to add Babel?

BMX6 is widely used in guifi.net, but it is unmaintained (last code commit `65cb0d5` is from 2018[9]) and some community networks that are using it are switching to Babel or BMX7. BMX7 is also unmaintained (last commits are from 2019[10]), but the aim of this project is to provide a distance-vector easy to use alternative to BMX6 and BMX7. On the other side Babeld[11] and its bird's implementation of Babel[12] are maintained.

Some communities are already choosing Babel, in general they go for the babeld openwrt package, which according to their users, it has some problems: (1) the CLI is very limited, (2) it is not easy to debug on it and (3) it has no WebGUI at all [13]. On the contrary hand, bird is a solid and lightweight program used in production for BGP in large scale deployments. Regrettably version 1 is unmaintained, last release of version 1 is from 2019[14]. In version 2 there is the opportunity to also use it in combination with Babel. Bird has a complete CLI and, by the completion of this project, would come with a functional WebGUI when using it through OpenWrt.

1.3 Relevant deployments

Babeld is used as a default configuration for LibreRouter[15], and it is used in at least a relevant deployment in Quintana Libre of around 60 nodes.

Bird2 openwrt package (without uci and luci) is being used in a small community network of 9 with good results[16].

1.4 About the upcoming bird3

On December 2024, a new bird3 package appeared in openwrt-routing repository[17].

Working on bird2 openwrt integration is still relevant for the upcoming bird3, in the official announcement[18] they said:

> There are some minor breaking changes in config and CLI, most notably unified route attribute names to the filter variant. We are

[9] https://github.com/bmx-routing/bmx6/commits/master

[10] https://github.com/bmx-routing/bmx7/commits/master

[11] https://gitlab.nic.cz/labs/bird/-/tree/master/proto/babel

[12] https://github.com/jech/babeld

[13] this is just a luci status page that at least is not working in openwrt 23.05.5 release https://github.com/openwrt/luci/tree/master/applications/luci-app-babeld

[14] https://bird.network.cz/?download

[15] https://gitlab.com/librerouter/librerouteros/-/blob/librerouter-1.5/configs/default_config

[16] http://dsg.ac.upc.edu/qmpcav/index.php

[17] https://github.com/openwrt/routing/commit/5998119c5530fbd4ab844edee3c4da0ea4fec924

[18] https://trubka.network.cz/pipermail/bird-users/2024-December/017973.html

> expecting to add a compatibility mode for the CLI. Anyway, it should be possible to reuse most of the configs and CLI scriptings from BIRD 2.

It's also good to stay on bird2, specially, if we want to be able to run it on legacy and very old devices that are in the network:

> The memory consumption has gone up significantly. We are still working on reducing the memory footprint and the next versions should be better in that.

1.5 Resources

Here are some of the resources to take it into account for the development of this project

1. Bird2 babel docs[19]
2. Bird2 bgp docs[20]
3. OpenWrt package bird1 OpenWrt[21], project's extra documentation[22]
4. User guide on babeld[23]
5. OpenWrt package babeld[24]
6. OpenWrt package of luci app for babeld[25]
7. Babel IETF standard[26]
8. [BSc] Interoperability between classic infrastructure and Libre-Mesh networks in Guifi.net. Gioacchino Mazzurco, 2015, Università di Pisa[27]
9. [MSc] LEDE Firmware optimization for wired deployments using BGP and BMX6 for routing by enhancing and extending Bird Daemon's configuration and UI integration. Eloi Caso, 2017, Universitat Oberta de Catalunya[28]

[19] https://bird.network.cz/doc/bird-6.html#ss6.1

[20] https://bird.network.cz/doc/bird-6.html#ss6.3

[21] https://github.com/eloicaso/bird-openwrt

[22] https://github.com/eloicaso/bgp-bmx6-bird-docn

[23] https://openwrt.org/docs/guide-user/services/babeld

[24] https://github.com/openwrt/routing/tree/master/babeld

[25] https://github.com/openwrt/luci/tree/master/applications/luci-app-babeld

[26] https://www.rfc-editor.org/rfc/rfc6126

[27] https://nuvol.exo.cat/index.php/s/cQfdHpCnLowRnWM/download?path=%2F2.%20Memoria%20activitats%20publica%2F2015&files=lime-guifi-interop-internship-report_2018-build.pdf&downloadStartSecret=ht5dehqmgof

[28] https://github.com/eloicaso/msc_dissertation/blob/master/dissertation.pdf

2 GSoC 2025 plan

As a student, I am excited to contribute to Google Summer of Code (GSoC) 2025 with this project.

2.1 Milestones

Here are the milestones to track the project completion

During GSOC Community Bonding

1. Blog post announcing GSOC starts
2. Blog post that summarizes end of GSOC Community Bonding Phase

During GSOC MidTerm

1. Bird2 bgp is configurable through UCI
2. Bird2 bgp is configurable through LUCI
3. Blog post that summarizes end of GSOC MidTerm

During GSOC Final

1. Bird2 babel is configurable through UCI
2. Bird2 babel is configurable through LUCI
3. Bird2 bgp and babel working at the same time in the same device
4. Blog post that summarizes end of GSOC Final

2.1.1 GSOC Community Bonding

Tasks for phase 1 (May 4 - 28), introducing the gsoc contributor to the project

1. Get familiar on OpenWrt buildroot usage for 23.x
2. Get familiar on how luci works
3. Get familiar on how ubus works
4. Get familiar on how babel routing protocol works
5. Get familiar on how bgp routing protocol works
6. Learn to develop a basic openwrt-routing package
7. Get familiar with existing codebase of bird1-openwrt in lua and shell
8. Get familiar with test frameworks that could fit well with openwrt environment
9. Document the entire process at least with a blog post

2.1.2 GSOC MidTerm

Tasks for phase 2 (May 29 - July 10), work mainly on bgp but start approaching babel

1. Integrate and use end to end tests for frontend and backend in the project, to improve code quality and ensure no functionality is lost (specially on the unstable code or most complex UX journeys)

2. Set up a small testbed for experimenting the routing protocols
3. Adapt configuration syntax changes from bgp from bird1 to bgp bird2
4. Upgrade bird1-openwrt code to reach a bird2
5. Do code maintenance tasks, minor bugfixes and improvements
6. First development iteration on uci and luci bird2 babel development
7. Document the entire process at least with a blog post

2.1.3 GSOC Final

Tasks for phase 3 (July 14 - August 21), work mainly on babel and finish project

1. Last development iteration on uci and luci bird2 babel development
2. Improve bird2 bgp and babel integration, ensure it will work for bird3
3. Refactor code if needed, so it is easier to maintain
4. Document the entire process at least with a blog post