Overview

This document is a general overview of the technical pieces associate with Toronto Mesh (tomesh). It is intended for those trying to learn more about tomesh, presenting at a high level the pieces that we hope one day will fit together into a community mesh network, and the current statuses on these initiatives.

A node is a bundle of free and open-source software running on off-the-shelf and reproducible low-cost hardware. These nodes, each one owned and operated by a user who peers with neighbouring nodes, form the mesh network. This network is analogous to our often privately-owned last-mile networks, except it is maintained by the community of users. We use the network for private and secure communication, including real-time use cases (e.g. instant messaging, voice and video chats) and hosting persistent services (e.g. a blog, a web application, peer-to-peer services).
Design Goals

- private and secure communication among nodes
- ownership of the network never becomes centralized
- resilient operations without central points of failure
- easily deployable and self-maintainable in other regions, not just Toronto (distribution of knowledge and accessible support channel are key considerations that affect technical roadmap)
- eventually have low-cost fast Internet access over the mesh network by a provider of the user’s choosing

Hardware

An Orange Pi Zero node attached to two Toplinkst TOP-GS07 WiFi radios in a 3D printed case.

Single-board Computer

This is the physical board of a tomesh node. The boards we are currently using are Raspberry Pi 3 or Orange Pi Zero. Both boards have strengths and weaknesses. Other boards can work as well but these are the two main ones that we actively test with.

Raspberry Pi 3

- Faster processor sends and receives encrypted traffic at about 80 Mbps
- On-board 2.4 GHz WiFi Access Point
- Available locally for about CAD 53

Orange Pi Zero

- Slower processor sends and receives encrypted traffic at about 66 Mbps
- On-board 2.4 GHz WiFi Access Point (weak signals and speeds < 10 Mbps)
- Smaller than the Raspberry Pi 3
- Fairly cheap board at CAD 11 + shipping cost from China (not locally available)

WiFi Radio

The radio is used to establish a wireless link to your neighbouring nodes. On-board wireless modules on both the Raspberry Pi 3 and Orange Pi Zero do not support the required protocols for peer-to-peer wireless links (i.e. Ad-hoc or 802.11s Mesh Point), so we use USB adapters to extend the functionality.
TP-LINK TL-WN722N

- 2.4 GHz radio, supports 802.11s Mesh Point at about 20 Mbps
- Detachable Antenna
- Available locally for about CAD 18

Toplinkst TOP-GS07

- Dual-band 2.4 GHz & 5GHz radios, supports 802.11s Mesh Point at about 45 Mbps
- Dual antennas 2T2R (can be detachable with board modifications)
- Small form factor
- Gets very hot and requires a fan for cooling
- Fairly cheap radio at CAD 9 + shipping cost from China (not locally available)

Software

We have two software projects that create a node with the above listed hardware:

- prototype-cjdns-pi
- mesh-orange

The first one is as the name states, a prototype, where we move fast and add many optional features to try them out. You can, for example, turn on IPFS for distributed content hosting on your node. There are detailed instructions on how to install this on your hardware, and you can hop on to our chat for help.

The other repository, mesh-orange, is designed for production use. It is a minimal operating system for the same hardware, where we give a lot more thought to consistent configurations and upgrade paths. This is currently a work-in-progress.

Almost all existing nodes are running the prototype, as our current network of 20 nodes is considered an early test net. The long-term plan is to transition to flashing SD cards with images of mesh-orange, rather than our current process of flashing Raspbian/Armbian then running prototype install scripts on top of that.

Peer-to-peer Wireless Link

We use the 802.11s standard to establish wireless links using the radio hardware on your node. The protocol keeps track of wireless connections within the radio's range, so all tomesh nodes that are reachable become peers. It is also possible to create a mesh using only 802.11s, but we have disabled routing due to scalability issues with a 802.11s mesh. So we are using 802.11s Mesh Point just to establish connections to immediate neighbours.

Finding radios that support this standard has been a challenge, and the Toplinkst TOP-GS07 is our most promising candidate.

Mesh with cjdns

For routing across the mesh network, we use cjdns. The philosophy behind cjdns is that networks should be easy to set up, protocols should scale up smoothly and security should be ubiquitous. Autonomous addressing and security guarantees are the primary reasons for choosing cjdns as our mesh software, but its end-to-end encryption of all network traffic also means we cannot run this on traditional routers that have very limited CPU capabilities. This is one reason why we run nodes on single-board computers.

cjdns is also a recent addition to the family of mesh networking protocols. While it addresses some pain points of other protocols, it also comes with its own set of technical challenges, such as poor routes in certain scenarios. There is also no existing large deployment of a physical cjdns mesh network (nodes being meshed over peer-to-peer wireless links). Existing cjdns networks are mostly formed by tunnelled connections over the internet to simulate a physical link. So tomesh may also serve as a test net to improve the cjdns project itself.
Addressing

cjdns uses a self-addressing scheme by using the hash of the node’s public key to assign an IPv6 address. The addresses on a cjdns network start with `fc` in the `fc00::/8` space and that is why sometimes it is called the `fc00 network`. Since the system is self-addressing, no DHCP server or any other central authority is required.

Interfaces

cjdns currently supports UDP and ETH interfaces. The first one is what we use to create links that are tunneled over the Internet, through the connection provided by our home ISP. This allows our node to peer with nodes outside of our wireless radio range. Of course, over time we hope to replace tunneled connections with physical links. The latter ETH interface uses Layer 2 ethernet frames and beacons, which is what our 802.11s adapters use to peer with locally reachable nodes. Encrypted traffic from your mesh node and connected devices flow through a virtual cjdns interface, then gets passed onto these UDP and ETH interfaces, to be sent out to your peers, and then get forwarded on across the mesh network.

IP Tunnel

This is a feature to allow cjdns nodes that do not have an ISP connection to tunnel Internet traffic through the mesh network, by choosing an Internet-to-cjdns gateway node. This requires a gateway node to be set up, and then configurations on the client side to elect to use that gateway, much like how you would use a VPN. We have this working and will soon incorporate this into the prototype release.

Monitoring

Nodes may voluntarily report metrics over HTTP, and anyone can run a server to scrape all that public data to track the status of the network. Both reporting (Prometheus Node Exporter) and the collection (Prometheus Server) have recently been added to the prototype repo and can be switched on with flags, along with the fancy Grafana graphing UI.

Currently we are working on automatic node discovery and drawing the metric data onto our tomesh node map.
Next Steps

Actively being worked on:

- Hardware design and board modifications (heating issues, external antennas)
- Add cjdns IP tunnel capability to tomesh nodes (in prototype)
- cjdns performance (routing, multi-hop bottlenecks, IP tunnel bottlenecks, optimizations on SBCs)
- Test net deployment with nodes peered over the Internet
- Improve network status monitoring and mapping (using test net)
- Android app to allow mobile devices to work as mesh nodes

Planned or proposed:

- Make mesh-orange image usable as a minimal mesh node
- Script deployment of common applications that can run on mesh nodes as local services
- Design reproducible long-range antennas
- Design solar power solution
- Technical workshop series at public libraries
- 5 GHz mesh network deployment at Maker Festival 2018