1. Introduction

3 months ago, I decided that what my homelab was missing was a sense of community. I already used ZeroTier to connect back to my network, and was familiar with the idea of darknets from looking into dn42 a long time ago, and thought it would be awesome to try building my own!

For the purposes of SIGHORSE, we needed to define a scope, and for carpanet, v1.0 was defined as the following:

- Connectivity between members
- DNS (shared management)
- Some media-based service
- Centralized authentication

1.1. Vision/manifesto

A community of homelabs, where people can expose services and collaborate on something a little different from what they'd normally work on

An enterprise-like network, with all core services hobbyisted out with open-source software

A land to share your creations and art, your movie collections or your websites, a testing ground for interesting technology

1.2. Timeline

borne as cloudly 🌩️, soon to be renamed carpanet 🐟 since cloudly already exists as an AI app– carpanet harkens back to the days of ARPANET, but fish.

…

2025-03-01 - carpanet ZeroTier network created

2025-03-04 - carpanet discord created

…

2025-05-18 - The first connection is established (we have a second member on the network)

…

2025-06-05 - projectcarpa.net registered

2025-06-07 - the first carpanet fishing expedition (group call)

2025-06-09 - libraria goes live

2025-06-18 - carpanet joined SIGHORSE

2025-06-22 - Blog - **https://ow.bearblog.dev/carpanet-ldap-broken-because-dns/**

2025-06-23 - Blog - **https://ow.bearblog.dev/carpanet-the-reference-architecture-for-routing-and-dns/**

…

2025-07-16 - carpanet Energizer UPS adventure

2025-07-21 - libraria connects to LDAP

2025-07-23 - ACME server is brought up

…

2025-08-01 - Chris forks Technitium for OIDC

2025-08-14 - Etherpad joins the party!

2. Architecture

2.1. Network

2.1.1. ZeroTier

The overlay network that kicked off the whole thing- NAT traversal and UDP hole punching mean that I don't need to expose anything to the internet to connect, and the support for layer 2 bridging means I can share IP space with my existing homelab (and even do things like Wake on LAN directly from my mobile device).

I prefer it to Tailscale, feels more baremetal and less flashy, and seems to give a little more control

For carpanet, the original vision was to simply connect all members on a shared layer 2, and use BGP/OSPF to route between member sites, but complexity quickly became a concern, as did the fact that not all members had the ability to easily add a new router to their network (limited hardware, limited knowledge, limited willingness)

For this reason, carpanet became a single /16 supernet, containing each of the members' /24s.

2.1.2. Opnsense

Pfsense seemed sketchy, trying to download the package required a checkout screen

Setup was fairly straightforward, although learning the terminology and menu structure took a little while

The features we use the most are virtual IPs, allowing putting multiple machines into the carpanet network without adding them into the ZeroTier network individually

IP alias seemed to work the best, but issues were encountered with ARP for the Etherpad addition for some reason, requiring the addition of a Proxy ARP Virtual IP for my /24 range.

2.1.3. Pi as Router

Chris instead opted to use a Pi as a router

For reasons (lack of hardware mostly)

And it required config. I used iptables to forward interface traffic. Traffic destined for the carpanet IP range was static routed from my other devices to the Pi. The Pi then forwarded that traffic from the physical interface to the ZeroTier interface.

Why we didn't use BGP / actual routing (see Challenges / Onboarding)

2.2. DNS

For carpanet v1, DNS was a requirement, but I didn't want to manage raw BIND, nor did I expect others to learn it, so we needed a DNS server with some kind of frontend.

There are a number of options out there, and we evaluated primarily Pihole, Technitium, BIND with Webmin, and PowerDNS with PowerDNS Admin

2.2.1. PiHole

Very common, and I have familiarity from using it in my home network already

In its favor is the cleanliness of the web GUI, the ease of management and intuitive design

However, Pihole is designed primarily as a DNS forwarder/blackhole, and as such it only supports lists of local DNS entries fed into the forwarder engine, rather than a full DNS server database. This would have caused issues down the line when it came to expanding, as there would be no support for DNS zone hierarchy or granular permissions.

2.2.2. PowerDNS with PowerDNS Admin

PowerDNS Admin could have worked, but hasn't been actively maintained and was too buggy

2.2.3. BIND with Webmin

Webmin was clunky and too generalized

Of all the options, this landed in second place, but it wasn't particularly close.

The UI isn't bad, but Webmin being so broad worked against this option, as we only needed a full-time DNS server without additional plugins

Currently, this acts as a secondary node for some of the primary DNS zones

2.2.4. Technitium

Recommended by some of our friends, this is a dedicated DNS server written in .NET, and has a very well made GUI

After deploying it for evaluation, it was clear this ticked all the boxes for us, and was the server we went with.

One drawback we came across was lack of support for centralized authentication, however Chris was able to use his existing experience with C# and .NET to fork its repository and implement support for OIDC SSO, which will allow us to manage permissions and logins from a trusted third server.

2.3. Identity

Speaking of centralized authentication, we needed an IdP to act as the cornerstone of the network and hold the user database.

We ended up layering OpenLDAP with Keycloak, building the initial database in OpenLDAP, and shifting to management and synchronization with Keycloak, which provides OIDC SSO.

2.3.1. LDAP

Used OpenLDAP

Setting up the schema was rough, so I got some help– building the schema and data out using lfif files was tricky, since I wasn't finding an easy guide, but I did find some apps which could connect to the database, and build out all the requisite data for me! It was also able to create the groups.

Needed a frontend

LDAP user manager - **https://github.com/wheelybird/ldap-user-manager**

Built out the schema for me, and handled group creation like a charm

2.3.2. OIDC

KeyCloak was picked for this. It is backed by LDAP. Keycloak utilizes “realms” to separate permissions. The Admin role in the master realm has permissions over all other realms. You can get fairly granular with permissions. You can allow users to setup different 2fa methods such as a passkey or OTP code. Certain passkeys (such as Android native) don’t work however when there is no external access.

2.4. Certificates

Step-ca is the Certificate Authority. It is setup as an ACME CA for automated certificates. The CA started as pebble for a proof of concept, after which Boulder was tried. Boulder did more than needed and wouldn’t start to due to some conflicting services on the same server (the laptop in a drawer). Boulder was sunset and step-ca was born as the Carpanet CA.

3. Services

Calibre-Web (libraria)

The first service

I chose this because I felt we needed something to justify the network’s existence, but a streaming server like Jellyfin would be pretty resource-intensive

I don’t have a ton of hardware, compute, or storage, so an ebook library came across as the perfect choice, requiring minimal processing, traffic, and storage relative to video

Etherpad - what this was written in!

Not my first time with npm, but my first time watching it pull in this many resources!

I love that open source has progressed this far

Grafana

Who doesn't love some good monitoring? Webhooks can be used to send alerts to Discord. This was originally built with SQLite (ended up with corrupted config and no easy way to fix the database). It was rebuilt on a postgres database. It also supports Keycloak authentication.

4. Hardware (might cut this)

Justin's Homelab

Proxmox hypervisors on a cluster of adopted machines

Raspberry pi

Chris' Homelab

A raspberry pi

This Pi acts as a proxy. Incoming proxy (NGINX) for folks to consume different webservices behind it (such as Step-CA, Grafana, and KeyCloak). For out going traffic, everything destined for the carpanet subnet has a static route pointing to the Pi, after which, all traffic is forwarded from the physical interface to the ZeroTier interface that is destined for carpanet (using iptables).

Old college laptop in a drawer

Most services run in Docker containers. The CA runs natively.

5. Network Diagram(s)

6. The Cool Challenges (if not mentioned above) (Lessons Learned / What would we do over if we had the chance)

Building OIDC for Technitium (carpaDNS)

Link GitHub issues

What was the process of solving this like?

This was an interesting project. It took several iterations. The first being an entirely hijacked login page with hard-coded values (no option for non-OIDC login). The second major iteration was the option to do login with native Technitium or OIDC (still hard-coded values). The third iteration allows input in the Administration settings in the GUI and saves the config to disk in an encrypted config file. There where also some nuances between the Windows dev environment and Linux/Unix that needed addressed when all was done and working.

Onboarding (simplifying joining, compared to other darknets)

How do I sell the idea to other nerds?

How did I make this easy?

Discord server as hub

Easy to join

Harder to organize and keep information arranged