CARP (and friends)

Common
Address
Redundancy
Protocol

Presented by Nick Holland

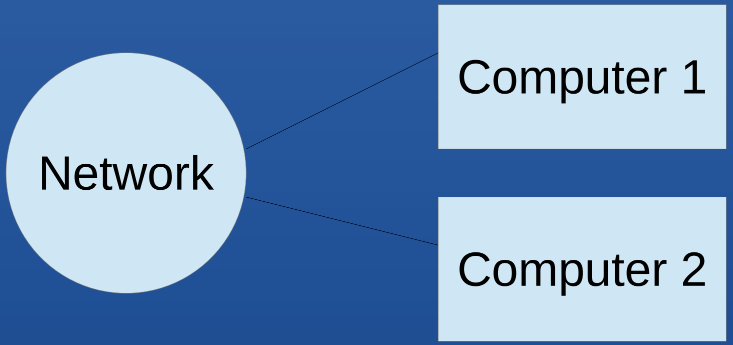
What is CARP?

- Allows multiple machines to provide “shared”, common IP addresses
- Load balancing
- Failover

- … and that’s it.

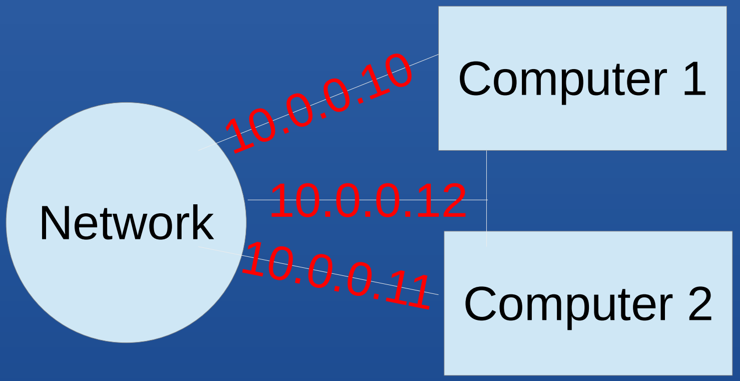
PhysicalView

- Two computers
- One physical NIC each



Logical view

- Computer 1: 10.0.0.10
- Computer 2: 10.0.0.11
- Shared CARP interface: 10.0.0.12



What CARP is NOT

- A tool to synchronize applications, data and state
- Keeps your configuration files in sync
- Magical redundancy for all things

Your link may change which machine it connects to, but the applications will (probably) be confused.

Your application will be back up “quickly”, but probably will break things for active users.

CARP isn’t magic

- **Some applications are very CARPable:**
  - DNS
  - NTP
  - Webservers serving small static(ish) pages
- **Some are not:**
  - servers serving “big” content
  - CGI/web applications
- Some are CARPable with helpers:
  - Firewalls (pfsync)

CARP: DNS

- Intrinsically redundant...but not really
  - Better term might be “Failure tolerant”
  - “failover” takes time (or...timeouts)
    - For each query
- Most of your assumptions about DNS failover are WRONG
- Works quite well with CARP, with good user benefit.
- Small network: better to have one CARP’d DNS server than several non-CARPed.

CARP: NTP

- Protocol is very effectively redunant.
- NTP works great with CARP...but why?
- If ONE NTP address or two NTP servers, CARP makes sense
- Multiple NTP servers, better to let NTP handle redundancy.
- AVOID the TWO NTP server configuration
  - Clients won’t will refuse to sync when two NTP servers disagree

CARP webserver

- Static pages: good
- Small applications without state: good
  - https://man.openbsd.org
  - https://cvsweb.openbsd.org
- Big, stateful applications: Provides rapid recovery only
- Big downloads: Provides rapid recovery only (probably)

Rapid recovery is good, of course.

CARP: DHCP

- Looks easy, it is not, at least with my expectations.
  - Machine X always gets the same IP address if possible
  - Many people are used to Common Reduced Anticipated Performance (C.R.A.P.) DHCP servers
  - OpenBSD dhcpd supports multi-machine synchronization
- DHCP lease database needs to be synchronized between nodes.
- You end up with multiple active DHCP servers on your network.

Firewalls

- CARP provides the common IP address
- PFSync keeps states in sync
- Without state, all active connections will put the *fail* in failover.
  - Review: State created on initial connection
  - Yes, state exists (to a firewall) for UDP.
  - Without knowing about an existing state, the second FW will just break the connection.
  - Without PFSync, you have rapid recovery, but broken connections.
- Real magic in a redundant FW is PFSync
  - But CARP is more fun to say

Importance of staying in sync

- CARP does nothing to synchronize system config. That’s your job.
  - Ansible (MAY NOT be the best answer!)
  - Scripts
  - IBS (but you missed that talk)
- Being out-of-sync can have uses!
  - test on secondary
  - or revert to unchanged secondary.

Things to keep in sync

- SSH host keys
- pf.conf
- SSL certificates
- Databases (dhcpd.leases, etc.)
- Users! (passwords, keys, home directories(maybe))
- Local scripts
- Crontabs
- …

Somethings are “one time”, others must be maintained

Things NOT to sync

- Some things may not be desirable to perfectly sync
  - Physical NIC config files (hostname.if)
  - Application configs (different IP addresses?)
  - …
- Strive to minimize these

Example 1: home FW

- DHCP from ISP
- Several subnets:
    - Home
    - DMZ/Wireless
    - Work From Home
- Desktop PCs, onboard (re) NIC, add-in 4-port 1G (bge) NIC
    - On-board used for external ISP – not always best choice.

Example: home FW part 2

- Redundancy where it counts
  - Redundant power supplies? Mirrored disks? WHY? Entire MACHINE is redundant!
  - (ok, I put in softraid mirrored drives, because I had a surplus of small Hds)
- ISPs requiring DHCP is a problem.
  - CARP doesn’t create a shared NIC, it creates a shared IP address.
  - DHCP takes place before IP.

    → thus, can’t request a DHCP config on a CARP address.

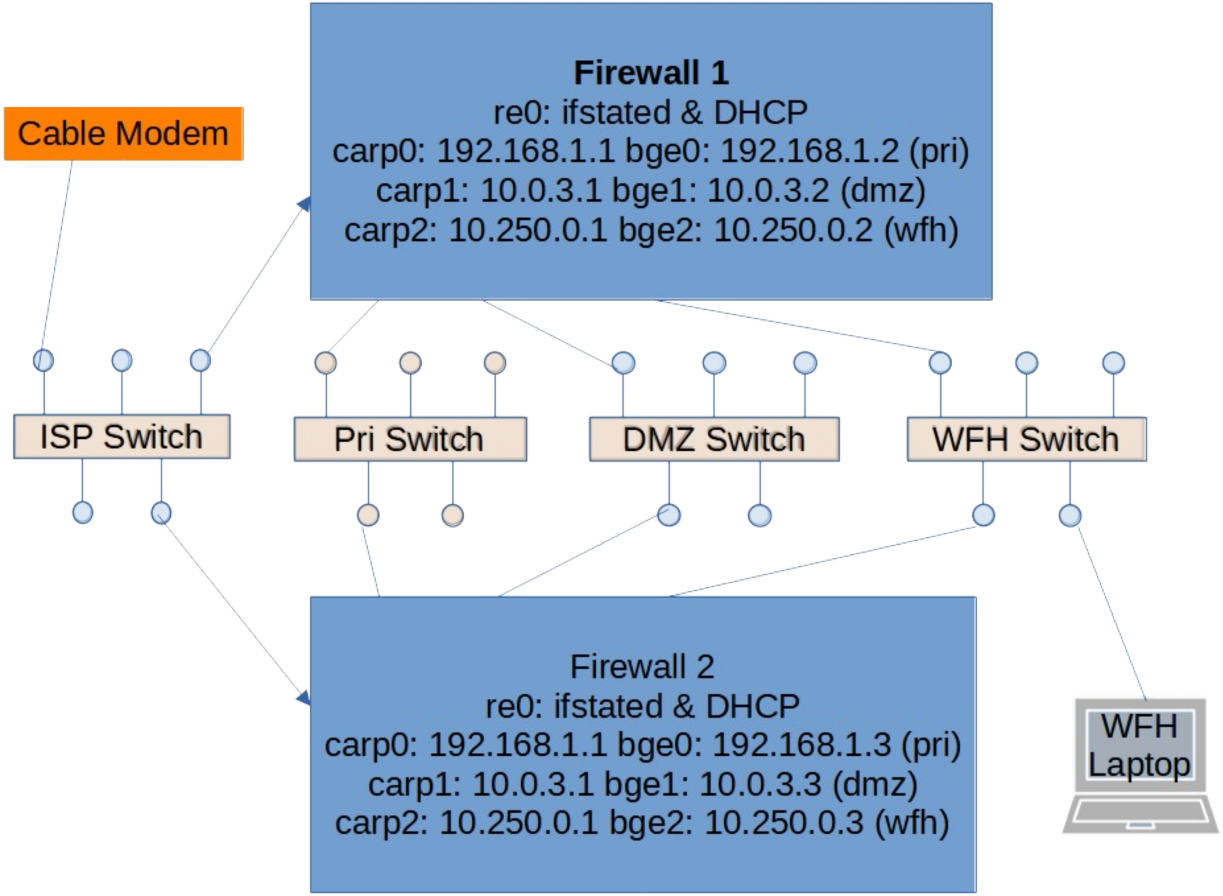
The DHCP problem

- ifstated to the rescue!
  - Monitors interface (NIC) status; “do stuff” when things change
- Configure ext NICs to have a fake MAC address, but not “up”
  - hostname.re0: `lladdr 00:00:1b:02:da:7f`
  - Both machines now appear to the ISP to be “same”.
  - But so far, neither is “active”
- External interface is NOT using CARP, but rather managed by ifstated.

Ifstated to the rescue

- When ifstated sees machine is “master” (active), it:
  - Flushes the routes
  - Requests DHCP config on the external interface (re0)
  - Reloads pf.conf
- When ifstated sees machine is “backup” (inactive), it:
  - Turns off external interface (re0): `ifconfig re0 -inet`
  - Flushes routes
  - Adds a new route through OTHER system (it still might need Internet!).

My Home Network



Firewall 1
re0: ifstated & DHCP
carp0: 192.168.1.1 bge0: 192.168.1.2 (pri)
carp1: 10.0.3.1 bge1: 10.0.3.2 (dmz)
carp2: 10.250.0.1 bge2: 10.250.0.2 (wfh)

Firewall 2
re0: ifstated & DHCP
carp0: 192.168.1.1 bge0: 192.168.1.3 (pri)
carp1: 10.0.3.1 bge1: 10.0.3.3 (dmz)
carp2: 10.250.0.1 bge2: 10.250.0.3 (wfh)

Files I need to worry about

- `/home/nick/*`
- `/etc/doas.conf`
- `/etc/hostname.re0, /etc/hostname.carp?, /etc/hostname.bge?`
- `/etc/pf.conf`
- `/etc/ssh/*`
- `/etc/ifstated.conf`
- `/etc/dhcpd.conf`
- `/etc/sysctl.conf`
- `/var/db/dhcpd.key`
- `/var/nsd/etc/nsd.conf, /var/unbound/etc/unbound.conf`
- `/var/nsd/zones/static.in.nickh.org`
- `/etc/hostname.pfsync0`
- *Whatever I forgot…*

Home FW experience

- ifstated – got help from Josh G. and Christer S. on misc@
  - Went together very well
- CARP: had problem with “carp0: incorrect hash” error.
  - Finally found I had used the same vhid on a set of machines I had used to test CARP config...and forgot was running. OOPS!
  - Had to add two small 5 port switches ($20 ea) for external and WFH networks, instead of direct wire before. Additional points of failure.
- All in all, not difficult, but VERY hard to justify the complexity for home use, except for practice.

Home FW build process

- Built up both machines physically
- Powered and networked both, then shut down one.
- Built CARP config on powered machine, moved network cables to new machine from previous firewall
- Got everything working properly, building a list of all files touched during the config (then rebooted without saving)
- Powered up secondary system, copied/adjusted files accordingly from first to second
- Setup PFsync, verified states replicated.
- Rebooted over and over, ‘cause it’s fun!

Example 2: OpenBSD mirror

- I run:
  - openbsd.cs.toronto.edu (install files)
  - obsdacvs.cs.toronto.edu (source code)
  - man.openbsd.org (with help) (man website)
  - cvsweb.openbsd.org (web access to CVS files and history)
- HW: Two very capable computers (replacing five capable computers, replacing two lame-*** but did the job systems)
- Sounds like a great job for a redundant CARP set. One live, one ready to go live.

Failover expectations:

- openbsd.cs.toronto.edu (install files, master for other mirrors)
  - Failover will break downloads
- obsdacvs.cs.toronto.edu (source code)
  - Failover will break downloads
- man.openbsd.org (with help) (man website)
  - Failover unlikely to be noticed
- cvsweb.openbsd.org (web access to CVS files and history)
  - Failover unlikely to be noticed

Toronto Mirror experience

- FORTUNATELY, thought of failure on day 1, all systems had same SSH ID keys (changing keys later is bad)
- Did NOT make a list of files customized per machine/task. Oops.
  - Took several iterations to get cvsweb running properly on both systems.
- anoncvs reposync doesn’t like changing sources, but is small, both systems update from master.
- cvsweb updates from anoncvs locally.
- Backup install system updates from live install file set (minimize upstream load and bandwidth)
- Upstream is IP restricted, so had to whitelist both physical (not CARP!) addresses

Mirror notes

- 36 files on my list of “are they in sync”
- SSL certs managed by acme-client
  - IF flipped every two or three weeks, certs take care of themselves
  - Really need a better process
- Problem with random flips.
  - Appears to be network timeouts.

Keeping things in sync

- External management:
  - Script on IBS backup server checks list of files for differences
- Internal management: Script run after change made to:
  - Generate diff between “this” server and “other” server
  - Show diff to administrator
  - Request explanation for changes
  - Store diff and explanation and user name of admin to file
  - Copy changed file to “other” systemm

Other notes

- Put your weak foot forward!
  - Your secondary system should be as capable OR MORE CAPABLE than the primary.
  - If your secondary system can’t do the job, you don’t have a backup.
  - If your secondary is better, and your primary can’t do the job, you have a quick, emergency upgrade.
- If you never use your secondary system, you don’t know if you have a backup.

OpenBSD encourages the use of the “preempt” option, where one node prefers to be the master. ***I do not.***

Resources:

- https://www.openbsd.org/faq/pf/carp.html
- https://man.openbsd.org/carp
- https://man.openbsd.org/pfsync
- https://marc.info/?l=openbsd-misc&m=167299046309551&w=2
  (guidance from Christer Solskogen on DHCP + carp)
- https://holland-consulting.net/scripts/ibs/ tool for backup and administration
- https://holland-consulting.net/scripts/remdiff.html Remote Diff
- https://egoslike.us/semibug/ my past and this SEMIBUG presentations

Questions?