Chapter 12

High Availability

A firewall device will eventually fail someday. And then it will not fulfill its most fundamental task of protecting computer networks.
Firewalls crash as much as other electronic gear. This constraint is an accepted fact and the reason why high-end firewall products have additional power supplies, fans, CPUs, or uplinks. Cheap consumer devices lack these add-ons so the best workaround is to deploy multiple devices that act as a group. This cluster of firewalls protects from the failure of a single device and provides high availability service.
Within the group, the devices agree that *one* firewall is doing the work and the other is just observing. Keeping track of the primary device is essential because the passive device must take over the operation if the primary firewall fails.

Basics

All firewall devices that join the *redundancy group* must comply with the standard *Common Address Redundancy Protocol* (CARP).
When CARP is enabled, the firewall will listen on its network adapters for vital signs of other CARP devices. The first member of the group considers itself the master and sends signs of life every second to the network. The second member of the group receives the keepalive packets and stays passive in a backup mode, which means: observe and wait.

Note

For CARP to work correctly, it is irrelevant if the group contains firewalls, routers, servers, or other gateways. This chapter uses these terms interchangeably.

As soon as the backup device misses three keepalive packets from the primary device, it must accept that the master has failed and considers itself the new master. It will then take over all tasks of the failed device as fast as possible so that end users will not recognize any outage.
Who is going to tell all computers on the attached network about the new firewall? Nobody does, because the new firewall also took over the IP address and MAC address of the redundancy group. For the other network device, nothing has changed at all – except for a brief interruption.
The life signs, heartbeats, or keepalives, are IPv4 packets sent to multicast address 224.0.0.18. These packets contain the virtual IPv4 address of the redundancy group that all CARP routers share. Also, each redundancy group has its unique group number, which makes it possible to operate several CARP groups in the same network segment.

The description of CARP and its functionality are very similar to the redundancy protocol *Virtual Router Redundancy Protocol* (VRRP). That's because most parts of CARP are a copy of VRRP. The license terms of the BSD operating system do not allow the use of VRRP, so the developers just coded a similar version and published it as CARP.

Lab network

The demonstration lab network contains three firewalls, two of which (RT-1 and RT-2) build a CARP cluster. Figure 12.1 shows the setup as a network diagram.
All subscribers of site 1 use as a standard gateway, neither of which is the IPv4 address of RT-1, nor that of RT-2, but rather the *additional* address 10.1.1.5 belonging to the CARP group. This setup belongs to the LAN side of the devices. On the WAN side, the firewalls create an additional CARP address along with their known IP addresses. For the failure protection

to work correctly, neighboring devices are required to communicate with only the virtual addresses of the CARP group.
On the WAN side, both CARP devices communicate with firewall RT-core, which becomes the target of many connection requests from location 1.

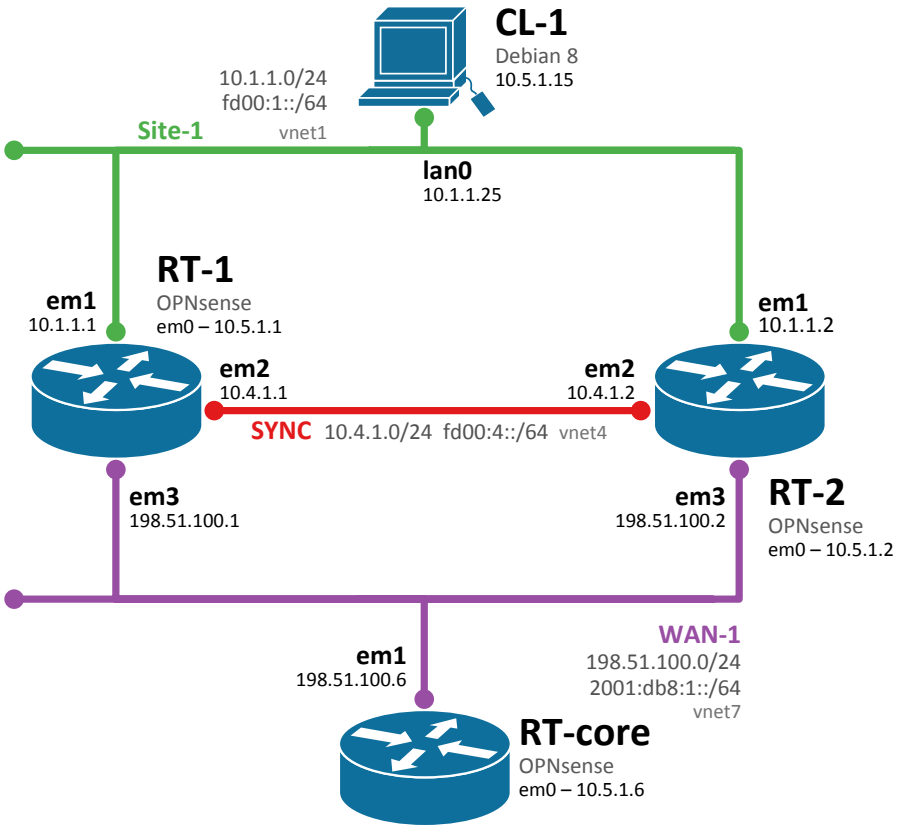

Figure 12.1: The firewalls RT-1 and RT-2 build a CARP cluster

The filter ruleset of all firewalls are less restrictive and permit everything so that network traffic can flow between all devices without hassle. Once all aspects of high availability are in place, the rules should be adjusted to enforce the local security policy.
The network segment SYNC is required and will be explained later in this chapter.

CARP group

OPNsense becomes a CARP router in the web UI at *Firewall* → *Virtual IPs* → *Settings*. Start the setup with the button *Add* and pick a unique group number and a virtual address with a suitable subnet mask. Table 12.1 lists the settings for the primary firewall RT-1.

Attribute	LAN	WAN
Mode	CARP	CARP
Interface	LAN	WAN1
Address	10.1.1.5/24	198.51.100.12/24
Virtual IP Password	*any*	*any*
VHID Group	1	1
Advertising Frequency	1/0	1/0

Table 12.1: Settings of the virtual IPv4 address

The *advertising frequency* has two meanings. First, it indicates the time (in seconds) between two heartbeat packets. Second, it reveals the priority of the sending firewall, whereby a lower value leads to a higher priority. An advertising frequency of 1 ensures that its sender is selected master. The other firewall needs a higher value to become the backup device.
The password protects the CARP group from unintended members and must be identical on all authorized devices.
When ready, hit the *Save* button and RT-1 will begin sending heartbeat packets on its LAN and WAN interfaces. Configure RT-2 similarly, with the exception of a higher advertising frequency, 1/10 for instance.
Several seconds later, both CARP candidates should have agreed on a boss (master) and an assistant (backup). In this example, RT-1 has won and assumed the master role (Figure 12.2).
Firewall RT-2 has similar settings, except for the status of BACKUP (Figure 12.3 on page 160). If RT-2 also shows MASTER, then both firewalls are in the master position and will fight for the virtual IP address. This misunderstanding should not happen during normal operation since it will lead to unexpected session disconnect on the client's computers.
Both firewalls will become MASTER if they miss the heartbeat packets of the other device. In that case, use the `ping` command to check if the machines can reach each other by the physical IPv4 address.

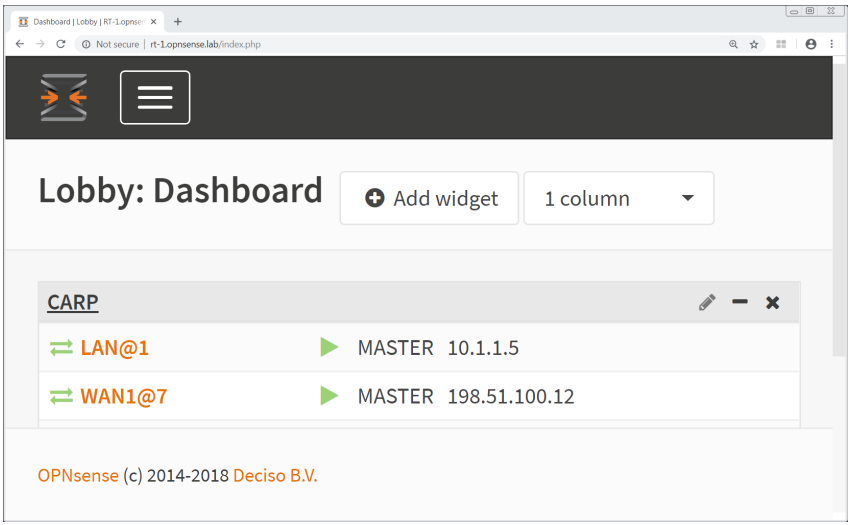

Figure 12.2: Firewall RT-1 becomes CARP Master

When MASTER and BACKUP reach an agreement, traffic can flow from client CL-1 through the firewalls to target host RT-core. Use `traceroute` to visualize the path through the network. The following listing shows that RT-1 is responsible for forwarding the traffic.

```
root@cl-1 ~> traceroute -I 198.51.100.6
traceroute to 198.51.100.6 (198.51.100.6), 60 byte packets
 1  10.1.1.1 (10.1.1.1)  1.362 ms  1.588 ms  1.562 ms
 2  198.51.100.6 (198.51.100.6)  2.097 ms  2.142 ms  2.448 ms
```

Now that everything is running smoothly, what happens when a device suddenly fails? Let's power off the master firewall RT-1 to simulate an outage and see what happens.
RT-2 no longer receives vital signs and promotes itself to master several seconds later. The same `traceroute` command on client CL-1 still reaches its target but shows a different hop in the path.

```
root@cl-1 ~> traceroute -I 198.51.100.6
traceroute to 198.51.100.6 (198.51.100.6), 60 byte packets
 1  10.1.1.2 (10.1.1.2)  1.240 ms  1.475 ms  1.458 ms
 2  198.51.100.6 (198.51.100.6)  1.573 ms  1.817 ms  1.854 ms
```

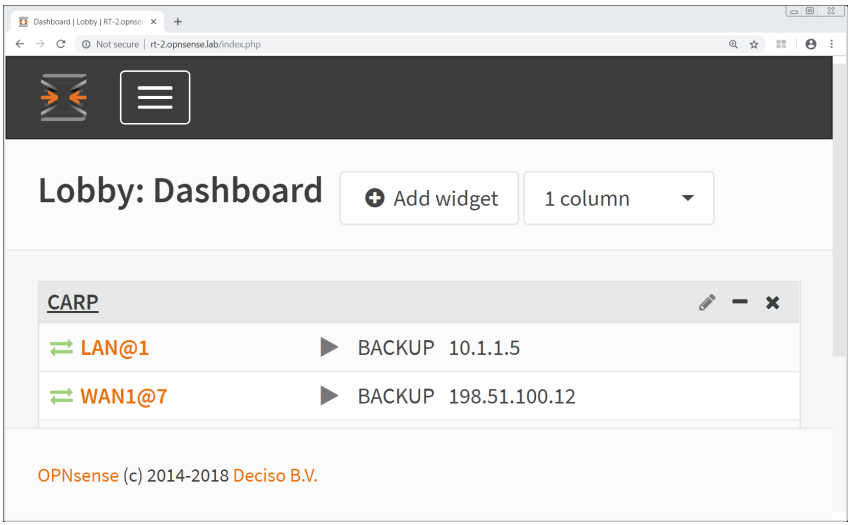

Figure 12.3: Firewall RT-2 becomes CARP Backup

Stateless

No client will notice the outage of firewall RT-1, unless it is sending or receiving data. So what happens during a file transfer?
The exact behavior of an interruption depends on the application and timeouts. A download on CL-1 from a public web server will stop during failover but continue after approximately four seconds.

At the moment, the configuration of all lab devices is somewhat unrealistic: the firewalls pass all traffic and also ignore the state of client sessions. Usually, corporate networks have limiting devices that use address translation (NAT) and strict firewall rules, which remember the state of each client connection.

Address translation

A router with Internet access usually has a packet filter enabled. Most likely it also modifies the IP address of transmitted packets (NAT, see Chapter 8) to replace private addresses by public IPs.

Let's move the lab network a little closer to reality and make the CARP devices translate addresses from the internal network 10.1.1.0/24 to a correct public IPv4 address. Figure 12.4 shows the setup of NAT on firewall RT-1.

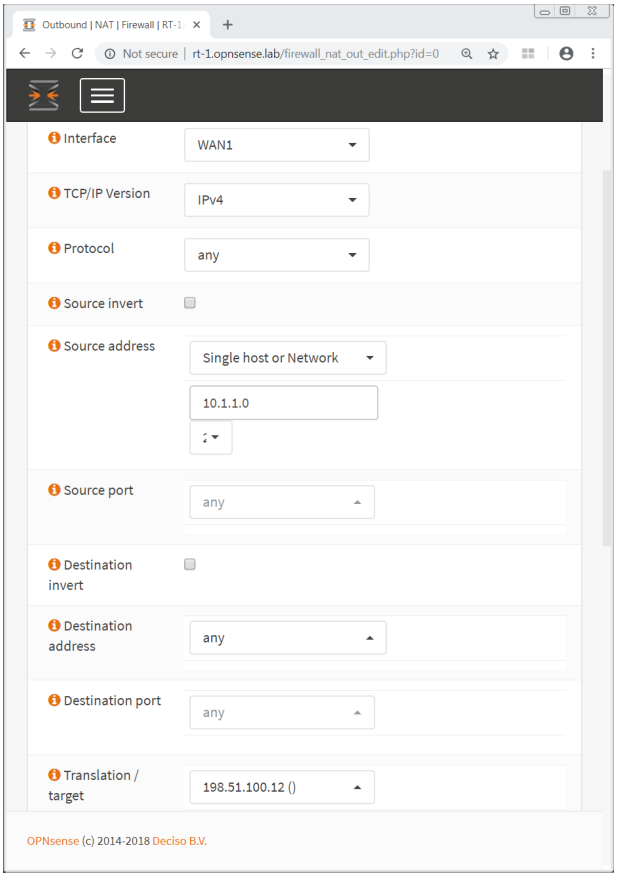

Figure 12.4: Firewall RT-1 translates IPv4 addresses

It is important to note that the OPNsense host should *not* use the IPv4 address of its WAN adapter (198.51.100.1), but uses the common CARP address (198.51.100.12) instead. The CARP address is present on all firewalls within the CARP group, so the address is still reachable if a single firewall fails.

It is best if RT-1 translates all outgoing packets into the CARP address, which will move to RT-2 during failover. That way, answering packets will flow back through RT-2 to the client.

Now the firewalls have to keep exact records: which packet must the packet filter accept, which IP is connected with which port, and how is the packet translated?
If the CARP master RT-1 fails, first, a data transfer on CL-1 will stop and break when the failover to the backup firewall is complete. The root cause of this behavior is the empty firewall table and empty NAT table in the backup device.

State tables

A state table is generally a security enhancement. It keeps track of all active sessions that the firewall has permitted. The packet filter and address translation process consult the session table to determine if the packet in question belongs to an established connection and may pass, or if the packet needs additional processing.
When using multiple firewalls, this leads to a problem because each firewall device maintains its local state table and is unaware of any other sessions. In a CARP setup, the master router handles all sessions and keeps them in its state table. The table of the backup router remains empty because this router has not seen a single connection yet.

Synchronization of sessions

OPNsense approaches the problem of differing session tables with the help of the operating system. For over ten years, FreeBSD includes the protocol *pfsync* for synchronizing packet filters. *pfsync* is independent of CARP, and together they make a good team.
Now the CARP master shares its session table knowledge to the backup router. The master sends all changes in its table to a pre-defined IPv4 address or multicast address. The backup router receives the messages and updates its local table accordingly. The objective is that all devices within the CARP group have the same content in the firewall table and NAT table. It is best to have a separate network segment dedicated to the state syn-

chronization. That's because synchronization is a real-time business. A session table dated ten seconds ago is not good for sessions that were established within the last nine seconds.

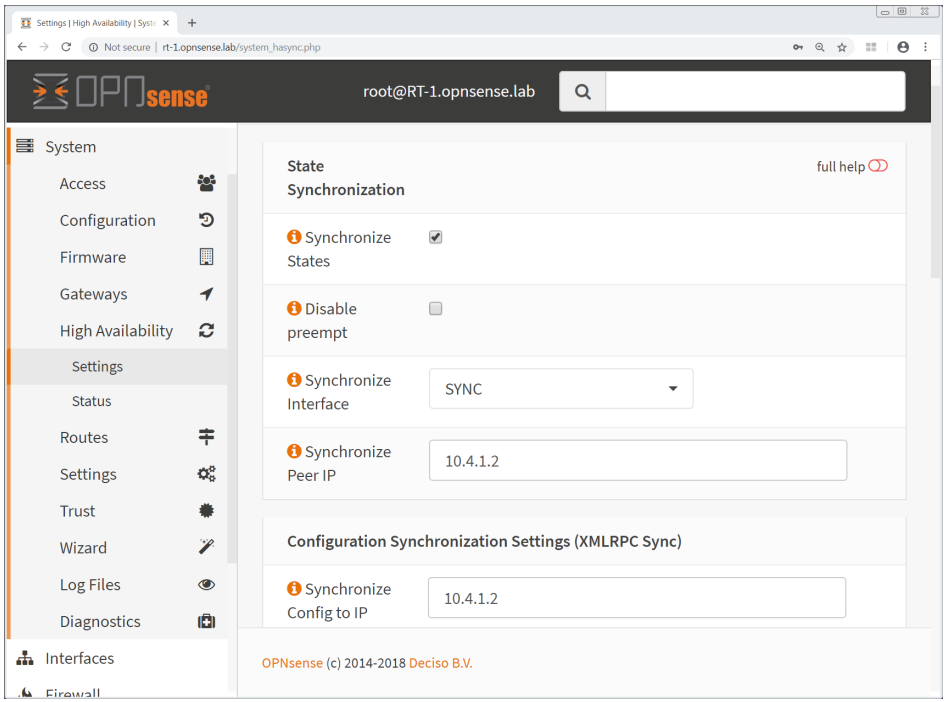

Figure 12.5: RT-1 relays all configuration changes to RT-2

Synchronization is configured at *System* → *High Availability* → *Settings*. The section *State Synchronization* expects a network adapter to perform the synchronization. This scenario uses the SYNC adapter and the IP range 10.4.1.0/24. The *Synchronize Peer IP* is the IPv4 address of the partnering router; RT-1 (10.4.1.1) will send all updates to the IPv4 address 10.4.1.2, which will finally reach RT-2 (Figure 12.5). The configuration of RT-2 is vice versa.

The *Synchronization Settings* are on the same web page and let the administrator decide which modifications are sent to the partner firewall. Section *Synchronization of configuration* will review the settings.

The exchange of table content runs over the interface *em2*, which provides a direct but independent connection between RT-1 and RT-2. In case there are two firewalls, it is best to use the target IPv4 address of the partner. When using multiple firewalls, leave the address field empty so that *pfsync* will use the predefined multicast address 224.0.0.240. Packets destined to that address will reach all receivers in the same network segment.

Now both CARP devices state each other's table contents. In principle, it would be sufficient if the backup router gets updates from the master, but synchronization runs in both directions.

Now, when it comes to an unplanned outage of the primary firewall RT-1, its partner RT-2 will take over the CARP role and start routing the client connections. That failover process takes a few seconds, but then any file transfer on CL-1 will continue because its session information is already present in the state table of RT-2 *before* the disaster happens.

```
root@RT-2:~ # /sbin/pfctl -s state | grep 10.1.1.25
all tcp 194.8.197.2:80 <- 10.1.1.25:41791 ESTABLISHED:ESTABLISHED
all tcp 10.1.1.25:41791 -> 194.8.197.2:80 ESTABLISHED:ESTABLISHED
```

Synchronization of configuration

OPNsense can do more than keeping the session tables in sync. The firewall is also able to replicate configuration changes from the primary host to the backup machine. Any changes to a firewall cluster need to be configured only on a *single* device. This consolidation saves time and prevents typing errors.

Luckily, there is no need to invent a new protocol for this task. OPNsense will execute the configuration change on both nodes at the same time. Communication with the remote firewall utilized plain HTTP. For that reason, setting up the *Synchronization Settings (XMLRPC Sync)* at *System* → *High Availability* → *Settings* requires a username, password, and the IP address of the secondary node.

Best practice

There are different methods for the redundant design of firewalls, which offer a smooth deployment and improve availability.

Asymmetric routing

If a packet takes a different path on the way to the server than on the way back, the routing is asymmetric. In theory, this is not a problem, but in practice, stateful firewalls, NAT gateways, or IDS systems prevent a successful connection.
It is easy with CARP to accidentally end up in asymmetric routing. This asymmetry even occurs in the laboratory network when RT-1 is master for the LAN side, and RT-2 is master for the WAN side.

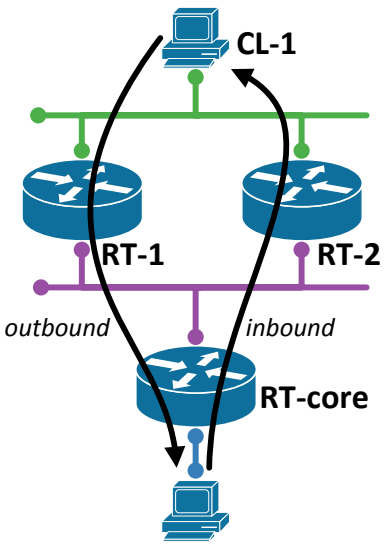

Figure 12.6: CARP might introduce asymmetric routing

In Figure 12.6 client CL-1 sends a packet to its default gateways, which is accepted by RT-1 as CARP master. The packet arrives at its destination via RT-core. On the return path, the packet reaches RT-core. It will pass the packet to the CARP cluster, and RT-2 receives it because it is master on the WAN side. However, RT-2 does not know anything about this connection

because it has not seen the initial packet, which was routed by RT-1. If RT-2 is just acting as a stateless router, it will relay all packets solely based on the routing table. But if RT-2 behaves *stateful*, it will dismiss packets that have no record in the session table. This behavior prevents asymmetric routing and successful communication of CL-1.
OPNsense can prioritize the CARP process so that the same firewall is master for all CARP groups. This election will render the routing symmetric. Assign priorities using the CARP advertising frequency, which is explained in section *CARP group* on page 158.

Master election

The CARP router with the highest advertising frequency will win the election and become master. The frequency is calculated from the *Base* and *Skew* values. A higher frequency sends keepalive packets more often. CARP then calculates the interval between two heartbeats with Formula 12.1. In a nutshell: the higher the numbers, the lower the advertising frequency. A lower advertising frequency ensures the firewall will not win the election to become the master router.

$$Interval = Base + \frac{Skew}{256} \tag{12.1}$$

If both candidates use the pre-configured values of Base=1 and Skew=0, then the router with the largest IP address will win. In this scenario, RT-2 becomes the master because its IPv4 10.1.1.2 is numerically larger than the IP 10.1.1.1 of its opponent RT-1. The same applies to the WAN subnet.
If RT-1 is the preferred firewall device and is chosen as master, then RT-1 must offer a better advertising frequency. Keep the Base value on both firewalls low (e.g. 1 second) for a quick failover and increase the Skew value of RT-2 to a high value of 100.
This modification will downgrade RT-2, and the master role swings from RT-2 to RT-1.

Synchronization

It is essential to choose the correct devices as source and destination for synchronization to prevent overwriting new content with old values.

The synchronization of state tables works bidirectionally. On both firewalls, insert the IP address of the partner node in the field *Synchronize Peer IP*. This task will even synchronize client sessions that are flowing over the backup firewall by accident.
The synchronization of configuration changes works unidirectional. The primary firewall executes the changes on the secondary firewall – and not in reverse. The field *Synchronize Config to IP* on RT-1 must be the IP address of RT-2. On RT-2 this field must remain empty. Any unintended changes on RT-2 stay there and will not endanger the stable operation of the primary firewall RT-1.

Quicker failover

Other protocols for redundancy and gateway availability achieve failover times below one second. The keepalive interval is within the range of several hundred milliseconds with a timeout of one second.
This indulgence timing is not possible with OPNsense. The predefined interval between two heartbeat packets is at the same time the minimum value: one second. Higher values are possible, but the timeout is always three times longer. By default, this is about 3–4 seconds.
A quicker failover is possible with CARP, but the software implementation of FreeBSD chooses a minimum value of one second. CARP can do a sub-second failover on OpenBSD, which is not the chosen platform below OPNsense.

Load balancing

CARP has *one* designated firewall that is actively working on the client sessions. Distributing the network load to several firewall devices is possible with a workaround.
This “poor man’s load balancing” (Figure 12.7 on the following page) assigns an additional CARP group to every network adapter. This group must elect the CARP backup of the primary group as master for the new group.
Following the example above, RT-1 is master and RT-2 is backup of the first CARP group. Within the second CARP group, RT-1 is backup, and RT-2 is

master. While group 1 serves IPv4 address 10.1.1.5, group 2 could provide address 10.1.1.6. The trick is to make half of the client computers use 10.1.1.5 as the default gateway, and the other half use 10.1.1.6 as their gateway.

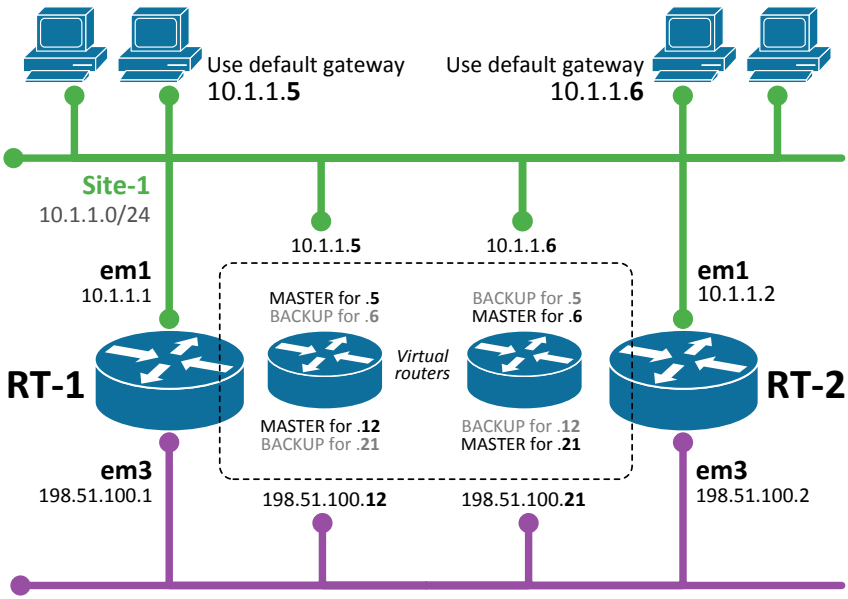

Figure 12.7: Load-sharing with CARP

Whether a firewall becomes master or backup depends on its advertising frequency, which is calculated from *Base* and *Skew*. The initial values are 1 and 0. The settings in Table 12.2 will promote RT-1 as master for group 1 and backup for group 2. The reverse settings apply to RT-2, which is backup for group 1 and master for group 2. Figure 12.8 shows the election result.

The exact numbers for the time offset are not critical, as long one firewall uses a higher value than the other.

Now both firewalls share the network traffic. The ratio of each device is not easily controlled: in a best case scenario, both gateways handle 50% of all packets, in a worst case RT-1 gets 99% of the sessions and RT-2 is almost idle with the remaining one percent.

Firewall	Interface	VHID Group	Base	Skew	Role	Address
RT-1	LAN	1	1	0	Master	10.1.1.5
RT-1	LAN	2	1	100	Backup	10.1.1.6
RT-1	WAN	7	1	0	Master	198.51.100.12
RT-1	WAN	8	1	100	Backup	198.51.100.21
RT-2	LAN	1	1	100	Backup	10.1.1.5
RT-2	LAN	2	1	0	Master	10.1.1.6
RT-2	WAN	7	1	100	Backup	198.51.100.12
RT-2	WAN	8	1	0	Master	198.51.100.21

Table 12.2: Advertising Frequency for load-sharing with CARP

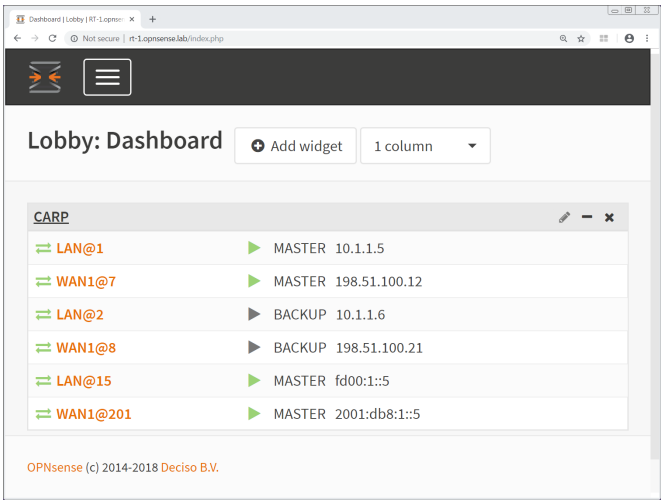

Figure 12.8: Firewalls RT-1 and RT-2 share the network traffic

IP version 6

CARP is ready for IPv6. But it needs more than that because the table synchronization has issues with the new IP version. The protocol *pfsync* is not yet migrated to IPv6. Unfortunately, the OPNsense web UI does not care and accepts an IPv6 address as input. But pfsync completely ignores the

strange address and keeps using the default IPv4 multicast address, even if the web page acknowledges an IPv6 address.
If the synchronization is performed over a separate network segment, nobody will notice the IPv4 connection, even if the remaining network is running purely IPv6.
Fortunately, the synchronization of configuration (XMLRPC Sync) is less stubborn and appears completely “IPv6 ready”.

Technical background

The web UI of OPNsense discloses many features and protocols that are used below the surface: CARP, pfsync, and XMLRPC.
CARP takes the user-defined *Virtual IP* and generates an additional IP address for the selected network adapter. This freely selectable IP address has the fixed MAC address 00:00:5e:00:01:*NN*, where *NN* represents the group number. This number is a unique identifier and permits multiple CARP groups to act in the same network segment without violating each other’s operation.
The vital signs from the CARP master are small packets targeted to the multicast address 224.0.0.18 or ff02::12 when using IPv6. This address allows all subscribers in the same network to receive the multicast packets.

The command line tool `ifconfig` shows all settings and election results. For instance, the LAN adapter of RT-2 reports about IP and CARP:

```
root@RT-2:~ # ifconfig em1
em1: flags=8943<UP,BROADCAST,RUNNING,PROMISC,SIMPLEX,MULTICAST>  [...]
     options=98<VLAN_MTU,VLAN_HWTAGGING,VLAN_HWCSUM>
     ether 00:15:16:02:01:02
     inet 10.1.1.2 netmask 0xffffff00 broadcast 10.1.1.255
     inet 10.1.1.6 netmask 0xffffff00 broadcast 10.1.1.255 vhid 2
     inet 10.1.1.5 netmask 0xffffff00 broadcast 10.1.1.255 vhid 1
     inet6 fe80::215:16ff:fe02:102%em1 prefixlen 64 scopeid 0x2
     inet6 fd00:1::2 prefixlen 64
     inet6 fd00:1::5 prefixlen 32 vhid 15
     nd6 options=21<PERFORMNUD,AUTO_LINKLOCAL>
     media: Ethernet autoselect (1000baseT <full-duplex>)
     status: active
     carp: MASTER vhid 2 advbase 1 advskew 0
     carp: BACKUP vhid 15 advbase 1 advskew 10
     carp: BACKUP vhid 1 advbase 1 advskew 10
```

CARP would not be complete without its synchronization buddy pfsync. The handling is very similar, and it also uses `ifconfig` as a configuration tool. However, pfsync is independent of an existing network adapter and even creates a new adapter with the appropriate name *pfsync0*. RT-2 can easily display adapter's settings:

```
root@RT-2:~ # ifconfig pfsync0
pfsync0: flags=41<UP,RUNNING> metric 0 mtu 1500
        groups: pfsync
        pfsync: syncdev: em2 syncpeer: 224.0.0.240 maxupd: 128 defer: off
```

No separate software process is required for the high-availability stuff – because the kernel performs these tasks on its own.

Summary

High availability of firewalls with CARP and pfsync is a stable and straightforward way of lowering the downtime of services. The trick is to use multiple firewalls: *two* identical devices build the firewall group of which the first device runs the business and the second device takes over as soon as its partner fails.
The additional configuration effort is minimal, and *XMLRPC Sync* will replicate the configuration changes to all members of the firewall cluster.