VIRTUALBOX

Credit: www.virtualbox.org

Get to grips with virtual networks

Part One! Don't miss next issue, subscribe on page 16!

Harness the power of VirtualBox networks to create self-contained virtual networks within your own computer. **Stuart Burns** reveals all.

OUR EXPERT

Stuart Burns is a Fortune 500 network administor specialising in virtualization at scale. When not doing that he can be found experimenting.

Out of the box, *VirtualBox* does a decent job of networking for general-purpose, short-lived VMs. It's capable of much more, though, including being able to create internet-routable test lab networks – even several test labs. They can all talk to each other and enable the owner to manage everything within that test network that you may not be able to manage on your locked-down ISP router, including self-managed DHCP, DNS, TFTP and more. Create a new, separate virtual network to experiment in is the answer. It can be used, destroyed and rebuilt as the reader sees fit, without breaking anything internet-related for other members of the household.

By default, when a virtual machine is created in *VirtualBox* it uses a NAT'd connection. This simply means the computer upon which the VM runs on is managing its connection on the VM's behalf, in what is in effect a network bubble with internet access. The downside is that other computers on your main network won't be able to talk to any VM you create on your computer with out-of-the-box NAT.

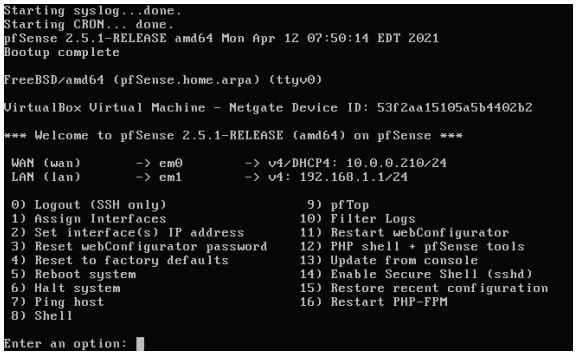
It's possible to change this and add it to your true local network by opening the VM network connection in *VirtualBox* and selecting Bridged mode on the network connection. Bridged mode puts the VM on the same network as the other laptops or desktops, ie the main network, sharing the local network and getting an IP address on that network from the local DHCP server. However, it would still be on the "main" network.

QUICK TIP

There's no reason why several networks can't exist at the same time, as long as the networks don't clash. The only item required is ensuring the DHCP range doesn't clash and routing is configured. Rather than recreate it all from scratch, use virtualbox to clone the pfsense network.

Configure your IP address

To build our isolated network the first thing to do is to make sure your *VirtualBox* workstation has a persistent IP address if possible (this saves a lot of work later). This is because the rest of the network has to know how to access the test network. An ever-changing IP would make life difficult. Most ISP routers enable you to statically map MAC addresses to an IP address, essentially making it a static IP address in all but name. The reader may need to refer to the router's manual. This step is important because we'll be telling all the machines on the local network that to access the isolated network, they need to send all the traffic to the *VirtualBox* router being created shortly to forward (pass on) the traffic. This is known as "routing".



Here's a cConfigured pfsense router with network setting for both WAN and LAN. If WAN is set to DHCP this can change – proceed with caution.

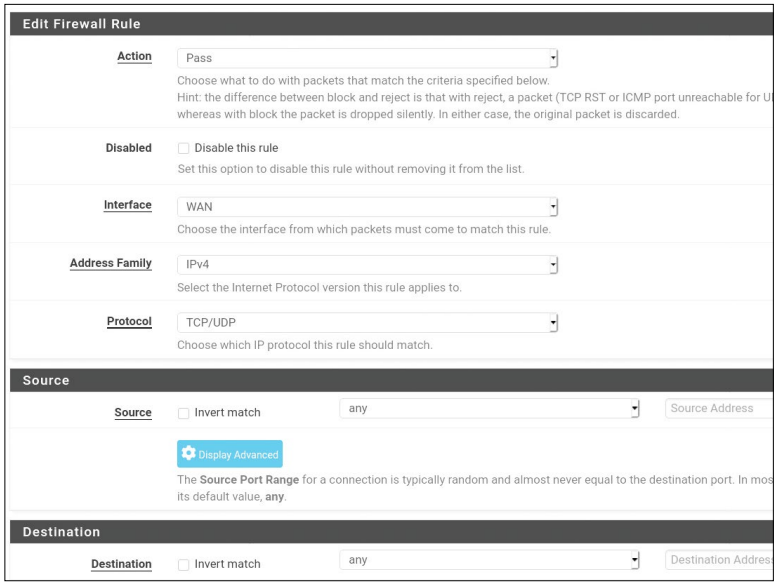
To manage the routing and firewall aspects this tutorial uses pfSense, a secure and well-regarded (but free) commercial firewall distribution. It can be downloaded from **https://nyifiles.netgate.com/mirror/downloads/pfSense-CE-2.5.1-RELEASE-amd64.iso.gz**

By default, the download is gzipped, so extract the ISO file to somewhere handy. Create a new VM. Pfsense is BSD so be sure to select 64-bit FreeBSD as the machine type in *VirtualBox* when creating the VM. Enable the first two network adapters on the VM. The requirements for FreeBSD are extremely modest: 512MB RAM and one vCPU with 8GB disk will be plenty for this scenario. It's important to set the first virtual network card to bridged (onto local network) and the second one to "Internal Network" and "intnet". Doing this sets one foot in both networks and pfsense enables us to manage the communication between the two. Before powering it on, attach the pfsense ISO that was previously downloaded.

After booting, run the installer. For the purposes of this tutorial just go with defaults all the way through and then shut down the VM. Remember to detach the installation media on completion. Once restarted the pfsense router should look something like that shown in the screenshot (*above*). Note down the LAN address because you'll need it later. You should also have a DHCP-assigned WAN address and LAN address. If either of these are missing then the isolated network won't work as expected and you'll need to revisit the previous steps.

The next job is to create a VM inside our isolated network. Create a Ubuntu (or other Linux flavour) desktop and set the network adapter to the *VirtualBox* network, intnet. All being well it should receive a DHCP address from the newly installed router because, out-of-the-box, the test lab side of our router will turn on DHCP for the internal network by default. It can be tweaked later. Open the web browser of the newly created VM and enter the IP address for the LAN gateway that was noted down before. Sign into the pfsense system using "admin" and "pfsense" in the first instance.

When the pfsense web page is opened for the first time it launches a wizard. Leave all the options as standard except taking the option to change the DNS address used. Doing this means you can use a privacy-respecting DNS service such as cloudflare (1.1.1.1) or quad9 (9.9.9.9) and at the same time configure the local domain details (if desired). On step four, unticking "Block RFC1918 Private Network" should be done if you want to modify the pfSense firewall rules from the main network. Finish the installation by pressing the "reload" button at the end of the wizard.



An example of how the WAN firewall should look to allow traffic in and out. This configuration allows everything in, so be cautious.

Connect to the web

Next, check that those virtual machines in the test network can connect to external websites. Use a web browser to try and browse an external website. Once confirmed there are a few tweaks to be made at the firewall level to make sure all traffic in and out can pass through. By default, all WAN traffic inbound is blocked. Open the pfsense VM webpage again and navigate to Firewall> Rules> WAN. Click the green upwards "Add" arrow below and change the following to allow any inbound traffic as shown in the screenshot (*above right*). A description is useful, but not required. Press Reload to implement the changes. Note: this rule set allows any traffic in or out. It isn't concerned with security. It can be tweaked, but for our internal infrastructure it's okay. Repeat the same but use the LAN interface.

Finally, tell the other computers on your real local network how to get into the isolated network. This is done with the `route` command. For each computer that needs access to the isolated network the reader needs to use the `route` command to set up the routing correctly. Depending on the operating system this may require a different layout, but the same command.

On a Linux desktop use the following (as root, or use `sudo`). The IP route add should be obvious enough and substitute the network in question for the LAN network we noted down earlier. Finally, the 10.0.0.55 in this example is the WAN interface, again, taken from the pfsense screen.

```
ip route add 192.168.0.0/24 via 10.0.0.55
```

On windows the command would be (again, running as administrator):

```
route add 192.168.0.0 mask 255.255.255.0 10.0.0.55
```

These routes tell the machine you're working on that any traffic that needs to be sent to the isolated network can be reached by sending traffic to the IP address 10.0.0.55. These route commands will last as long as the machine is up and will be lost on reboot. To make the route permanent, the administrator will need to refer to the documentation for their current OS as implementation can vary between OS implementations.

If needed, to remove the route use the command with the delete switch, for example, `route del 192.168.0.0`. If the pfSense WAN address keeps changing you'll need to remove the old route and add one with the new address. This is why the mapping is required. This process will also need to completed on all machines that need to reach the lab. Now the administrator should be able to SSH, RDP, FTP or such into your test lab from your local network. LXF

» LET'S TALK FIREWALLS

While what we've done could actually be achieved using standard routing on a Linux router, the use of a firewall and routing combination allows for more fine-grained access control. For example, it makes it possible to restrict inbound or outbound traffic according to our own requirements. This can be useful. An example could be restricting inbound and outbound traffic to just HTTP/HTTPS or whatever port and protocol is required.

Indeed, when we add additional networks in the next installation we can fine-tune the access between the networks. Part of the effectiveness of the whole virtual infrastructure is that it can be as simple or as complex as required. The configuration used here is wide open however, and essentially there's no restriction between networks so anything on the test network can connect with anything anywhere. That's okay though because the network is only internal. Don't use such a configuration on the perimeter network as it will certainly ends in tears.

Note that if the BOGON network is unticked (as mentioned earlier) then it will allow the management of the firewall from the local network by entering the WAN interface and logging in using the pfSense username and password used earlier.