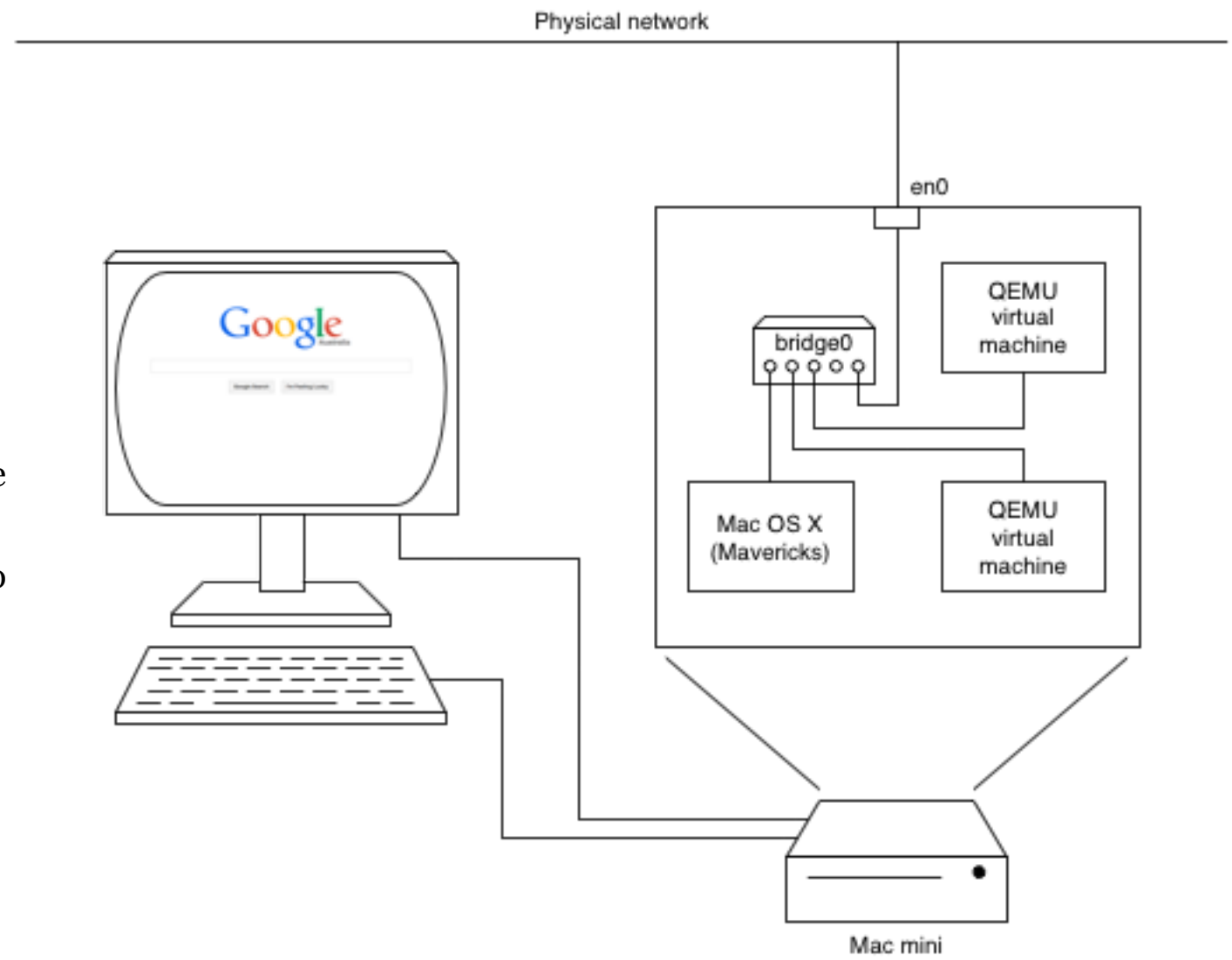
Networking bridging on Mac OS X (Mavericks) with QEMU

QEMU's user-mode networking is sufficient for many of the networking requirements of a typical virtual machine (VM) but does have some shortcomings:

1. The VM cannot have its own IP address. All network traffic to and from the VM goes through, and has the network address of, the host computer,
2. Pinging from the VM doesn't work due to the nature of the network implementation,
3. Inbound connections to the VM must be directed to the address of the host machine and then forwarded through to the VM by virtue of "hostfwd" parameters to the "-net user" option when QEMU is started, and
4. Forwarded ports must be configured when QEMU is started. They cannot be changed except by restarting.



A solution to these problems is to use bridged networking instead of user-mode networking for connecting the VM to the network. The diagram to the right illustrates the principle.

The diagram shows a Mac mini connected to a physical network using `en0`. The box above the Mac mini in the diagram represents the networking configuration inside Mac OS X. Instead of the operating system talking directly to `en0`, it connects to `en0` using a bridge. A bridge is a device which allows multiple network devices to intercommunicate. It is similar, but not the same as a hub. In this case the bridge is a virtual device and it can be created and configured via Mac OS X's Network Preferences pane.

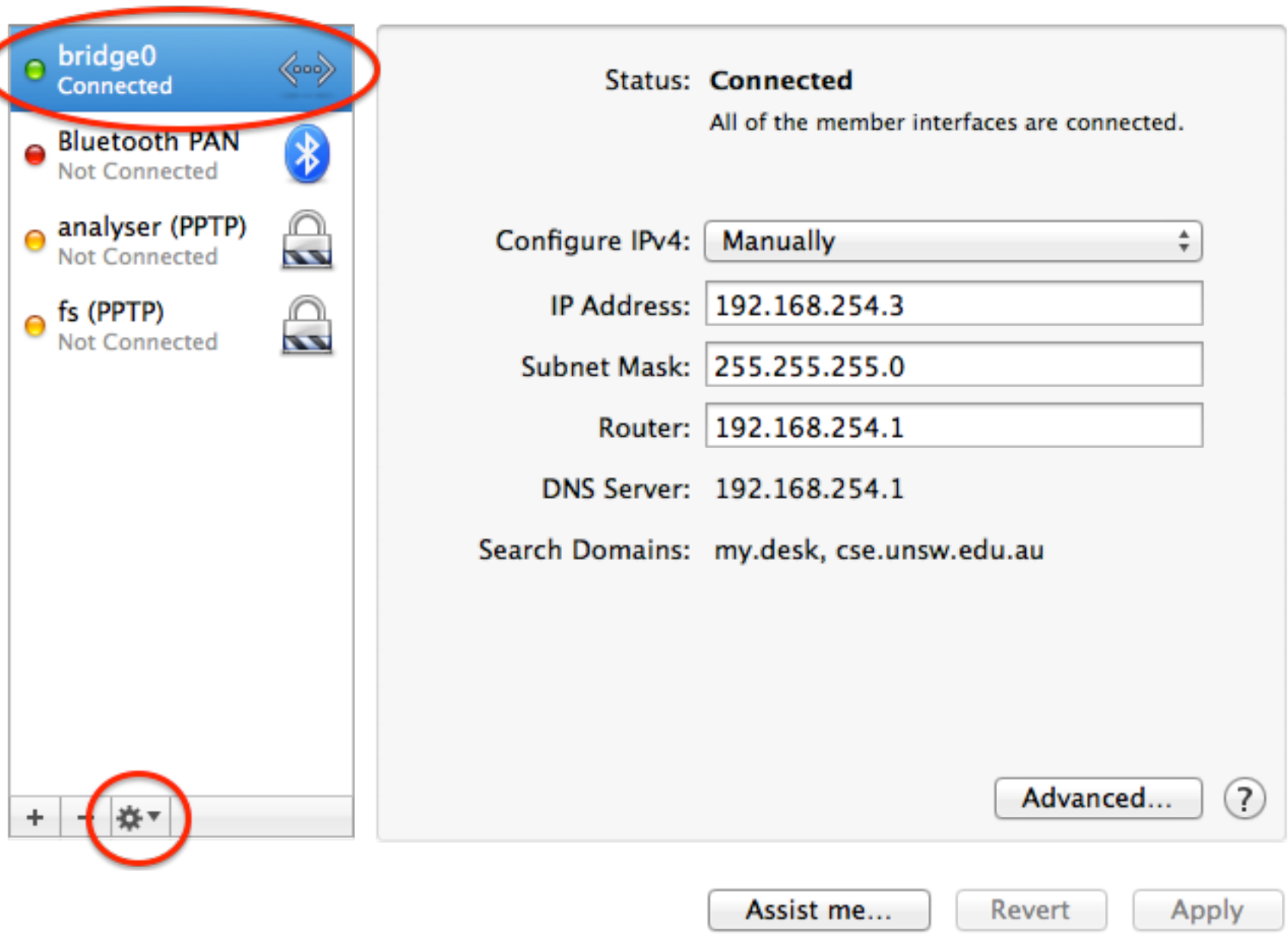
QEMU, when it is run to instantiate a VM, creates a TAP device through which the VM sends and receives its network traffic. The TAP device is virtual and behaves similarly to an Ethernet interface with its own MAC address. The TAP device connects virtually to the bridge. The bridge allows traffic from the TAP device to be sent straight out to the physical network, and allows network packets from the physical network to be passed directly to the TAP device. This can all happen mostly without Mac OS X's involvement so that it seems that the VM is actually connected directly to the physical network itself, with its own MAC address and IP address.

Each QEMU/VM has its own TAP device and this is created and attached to (or associated with) the bridge when QEMU starts, and is destroyed when QEMU exits.

There are caveats with operating a VM with bridged-mode networking:

1. QEMU must be started as the super-user (or administrator) like this: `sudo qemu <parameters>`,
2. The virtual bridge device must be created via Mac OS X's Network Preferences pane before running QEMU. However, this only needs to be done once for all VMs and it persists across restarts of Mac OS X.

Creating the bridge device



The diagram shows the Network Preferences pane from System Preferences on Mac OS X. Instead of having a physical network device (`en0`) in the list of interfaces, we see `bridge0` (circled). This is a virtual interface and we create it by first clicking the cog button on the bottom left of the pane (circled). We then select "Manage Virtual Interfaces...", add a "New Bridge...", associate it with `en0` and give it a name.

This new interface takes the place of our physical interface in the preferences pane and we configure its address and other parameters just as we would for the physical interface.

Configuring QEMU

To start QEMU with bridging, something like this is what you need:

```
qemu-system-x86_64 \
-smp 2 \
-vnc :5 \
-m 1024 \
-drive file=disk.img,if=virtio \
-device virtio-balloon \
-boot c \
-net nic,model=virtio,macaddr=54:54:00:55:55:55 \
-net tap,script=../scripts/tap-up,downscript=../scripts/tap-down \
-daemonize
```

The `-net` lines are the relevant ones.

The first `-net` option configures the network device's type and MAC address. The MAC address is important here because this is the address the physical network sees for the VM. It must be set explicitly if you use more than one VM because it always defaults to `52:54:00:12:34:56` and having more than one network host with the same MAC address worketh poorly.

The second `-net` option creates the network interface on the host side as a TAP device. The `script` and `downscript` parameters assign scripts which are run after the TAP device is created by QEMU when it starts and before it is destroyed by QEMU when it exits. In our bridging case here, they attach the TAP device to the bridge and remove the TAP device from the bridge, respectively. See below for example scripts.

Once a VM is running, you can use the command `ifconfig <bridge-device>` to see how the bridge and TAP device(s) have been configured.

tap-up

The following script, when provided as the `script` parameter to the `-net bridge` option, will attach the created TAP device to the bridge.

```
#!/bin/sh
#
TAPDEV="$1"
BRIDGEDEV="bridge0"
#
ifconfig $BRIDGEDEV addm $TAPDEV
```

tap-down

The following script, when provided as the `downscript` parameter to the `-net bridge` option, will detach the created TAP device from the bridge.

```
#!/bin/sh
#
TAPDEV="$1"
BRIDGEDEV="bridge0"
#
ifconfig $BRIDGEDEV deletem $TAPDEV
```

Tags:
network macosx qemu