In many instances, the lab designer will want to define a different default route for a container. The start.config definitions for each container include an optional `LAB_GATEWAY` parameter that, if set, will replace the default Docker gateway with the given gateway, and it will replace the resolv.conf entry and delete the route to the `my_host` address. That configuration setting is implemented using a *set_default_gw.sh*, which designers can optionally chose to directly use instead of `LAB_GATEWAY` in order to get more control over the setting of a default gateway, e.g., as part of parameterization. This script will automatically retain a route table entry so that the student can reach the "my_host" address. Additionally, those baseline images include a *togglegw.sh* script that the student can use to toggle the default gateway between one that leads to the host, and one defined for the lab. This allows students to add packages on components having lab-specific default gateways.

8.1 Network Taps

In general, Docker containers will only see network traffic addressed to the specific container, (or broadcast traffic). The behavior is consistent with use of a layer 2 network switch to interconnect containers on the same subnet. In some labs, the designer may wish to provide students with copies of all network traffic that occurs on one or more subnets. Labtainers supports network taps through use of two container base images: *tap* and *netmon*. The `tap` component should not be visible to the student, it exists to collect traffic off of all networks whose `start.config` definitions include the `TAP YES` attribute. The `netmon` component should be defined with a single network interface to a network called `TAP_LAN`. The `netmon` component should be the only one on the `TAP_LAN` network, (do not add the `tap` component to any network). The `tap` component must have the `TAP YES` attribute. A service runs on the `netmon` component that will receive network traffic sent by the `tap` component, and store it into the `/taps` directory within PCAP files named using the network name. See the `plc-traffic` lab as an example.

The `netmon` base is derived from the wireshark base. You may add other tools to that container as needed.

All containers attached to tapped networks will not be started until the tap and netmon containers are up and ready. This ensures that all startup traffic is captured in the PCAPs.

8.2 Realistic Network Routing and DNS

Some labs will strive to represent realistic networking environments, e.g., several networked components including gateways and DNS servers. To achieve that, you must override Docker, which automatically sets the container's /etc/resolv.conf file to use the host system DNS resolution. This is in addition to the default routes described above. While convenient, these mechanisms can distract and confuse students, particularly when routing and DNS resolution are central to the point of the exercise, (e.g., a DNS cache poisoning lab).

These Docker defaults can be easily overridden to present a more realistic networking environment. A worked example of such a topology can be seen in the *routing-basics* lab. This lab includes the following properties that can be reproduced in other labs:

- Default routes to gateway components.
- DNS definitions in /etc/resolv.conf that name gateway components.
- Use of `iptables` in gateway components to implement NAT.
- A hidden ISP component that exchanges network traffic with the host Linux system, thereby allowing all visible components to include routing, DNS and iptables entries that do not expose virtual networking tricks. See section 14.6 for additional information.