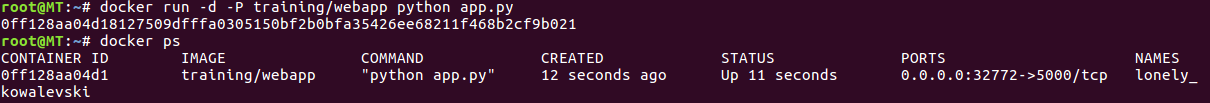
Docker网络

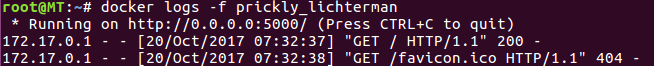
当使用 -P 标记时，Docker 会随机映射一个 49000~49900 的端口到内部容器开放的网络端口。

sudo docker run -d -P training/webapp python app.py



可以通过 docker logs 命令来查看应用的信息

sudo docker logs -f prickly\_lichterman



容器的连接（linking）系统是除了端口映射外，另一种跟容器中应用交互的方式。该系统会在源和接收容器之间创建一个隧道，接收容器可以看到源容器指定的信息。

使用 --link 参数可以让容器之间安全的进行交互。

sudo docker run -d --name db training/postgres

sudo docker run -d -P --name web --link db:db training/webapp

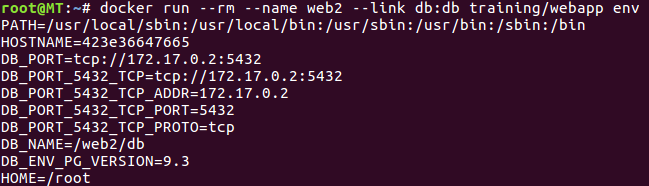
python app.py

--link 参数的格式为 --link name:alias ，其中 name 是要链接的容器的名称，alias 是这个连接的别名。

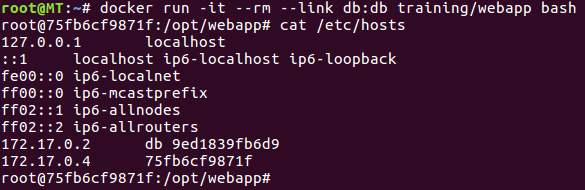
Docker 通过 2 种方式为容器公开连接信息：

1. 环境变量
2. 更新 /etc/hosts 文件

使用 env 命令来查看 web 容器的环境变量



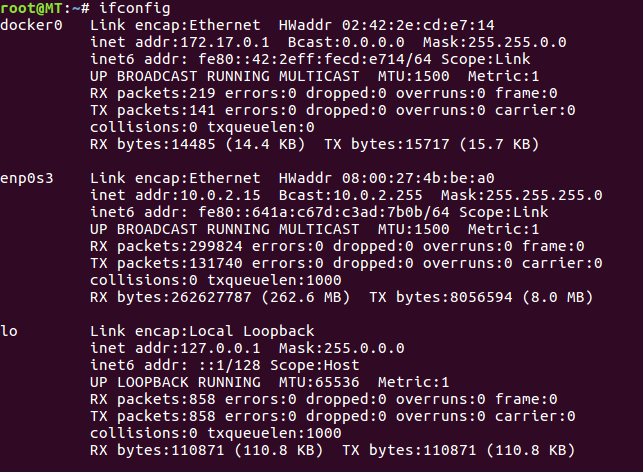
其中 DB\_ 开头的环境变量是供 web 容器连接 db 容器使用，前缀采用大写的连接别名。



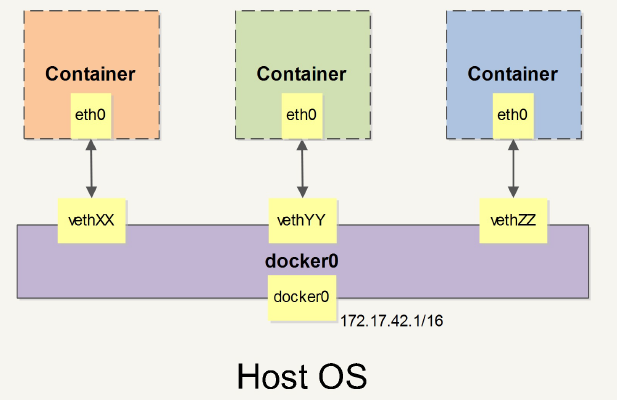
这里有 2 个 hosts，第一个是 web 容器，web 容器用 id 作为他的主机名，第二个是 db 容器的 ip 和主机名。

用户可以链接多个父容器到子容器，比如可以链接多个 web 到 db 容器上。

当 Docker 启动时，会自动在主机上创建一个 docker0 虚拟桥，实际上是Linux 的一个 bridge，可以理解为一个软件交换机。它会在挂载到它的网口之间进行转发。



当创建一个 Docker 容器的时候，同时会创建了一对 veth pair 接口（当数据包发送到一个接口时，另外一个接口也可以收到相同的数据包）。这对接口一端在容器内，即 eth0 ；另一端在本地并被挂载到 docker0 网桥，名称以 veth 开头（例如 vethAQI2QT ）。通过这种方式，主机可以跟容器通信，容器之间也可以相互通信。Docker 就创建了在主机和所有容器之间一个虚拟共享网络。



其中有些命令选项只有在 Docker 服务启动的时候才能配置，而且不能马上生效。

-b BRIDGE or --bridge=BRIDGE --指定容器挂载的网桥

--bip=CIDR --定制 docker0 的掩码

-H SOCKET... or --host=SOCKET... --Docker 服务端接收命令的通道

--icc=true|false --是否支持容器之间进行通信

--ip-forward=true|false --请看下文容器之间的通信

--iptables=true|false --是否允许 Docker 添加 iptables 规则

--mtu=BYTES --容器网络中的 MTU

下面2个命令选项既可以在启动服务时指定，也可以 Docker 容器启动（ docker

run ）时候指定。在 Docker 服务启动的时候指定则会成为默认值，后面执行

docker run 时可以覆盖设置的默认值。

--dns=IP\_ADDRESS... --使用指定的DNS服务器

--dns-search=DOMAIN... --指定DNS搜索域