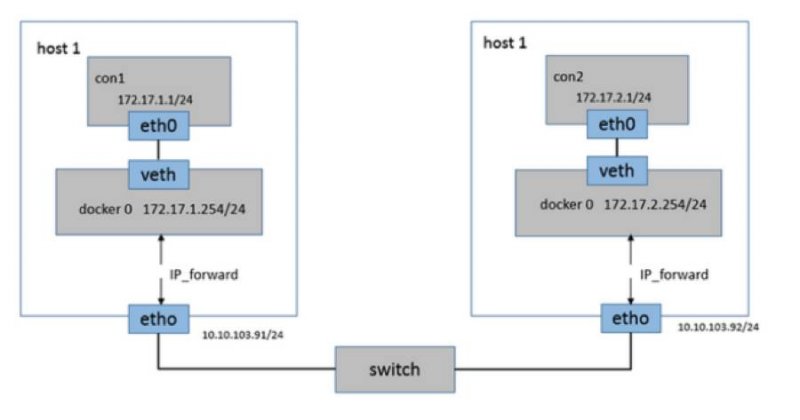
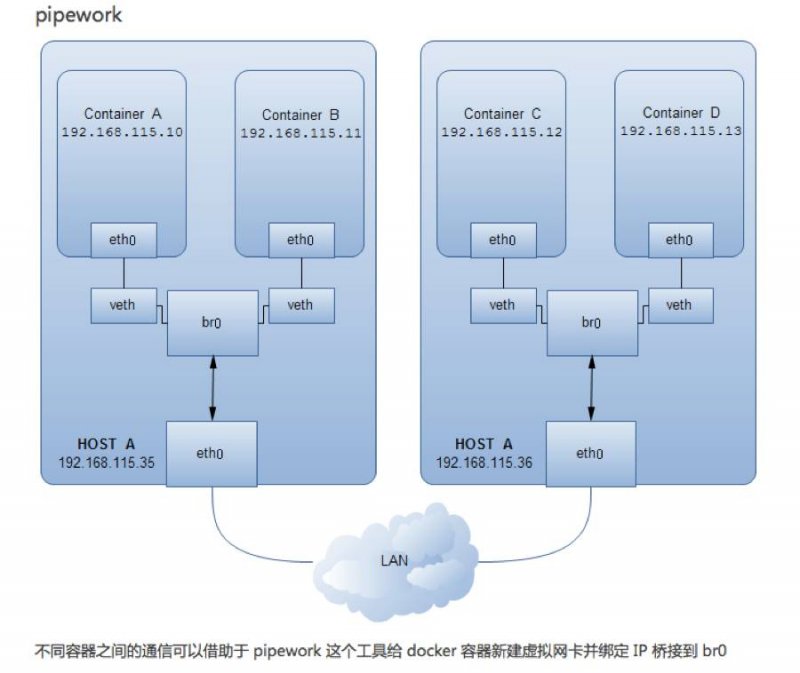
# 跨主机通信

Docker默认的网络环境下，单台主机上的Docker容器可以通过docker0网桥直接通信，而不同主机上的Docker容器之间只能通过在主机上做端口映射进行通信。这种端口映射方式对很多集群应用来说极不方便。如果能让Docker容器之间直接使用自己的IP地址进行通信，会解决很多问题。按实现原理可分别直接路由方式、桥接方式（如pipework）、Overlay隧道方式（如flannel、ovs+gre）等。  
  
直接路由  
http://cdn.img2.a-site.cn/pic.php?url=aHR0cDovL21tYml6LnFwaWMuY24vbW1iaXovUVAwQVk3dGRKblV4eFJNWjRRcDl0b21GaFFRMDNYVUUzS1BRMUx0VnE3b3JvTU1BSGxSMXo3ZXhLdUtUaWJjYmVpYzdBaWFWRDNNTU1KUW1uTWlhS2t5WXlBLzA/d3hfZm10PXBuZw==  
  
通过在Docker主机上添加静态路由实现跨宿主机通信：  
  


#### Pipework

http://cdn.img2.a-site.cn/pic.php?url=aHR0cDovL21tYml6LnFwaWMuY24vbW1iaXovUVAwQVk3dGRKblV4eFJNWjRRcDl0b21GaFFRMDNYVUUzS1BRMUx0VnE3b3JvTU1BSGxSMXo3ZXhLdUtUaWJjYmVpYzdBaWFWRDNNTU1KUW1uTWlhS2t5WXlBLzA/d3hfZm10PXBuZw==  
  
Pipework是一个简单易用的Docker容器网络配置工具。由200多行shell脚本实现。通过使用ip、brctl、ovs-vsctl等命令来为Docker容器配置自定义的网桥、网卡、路由等。

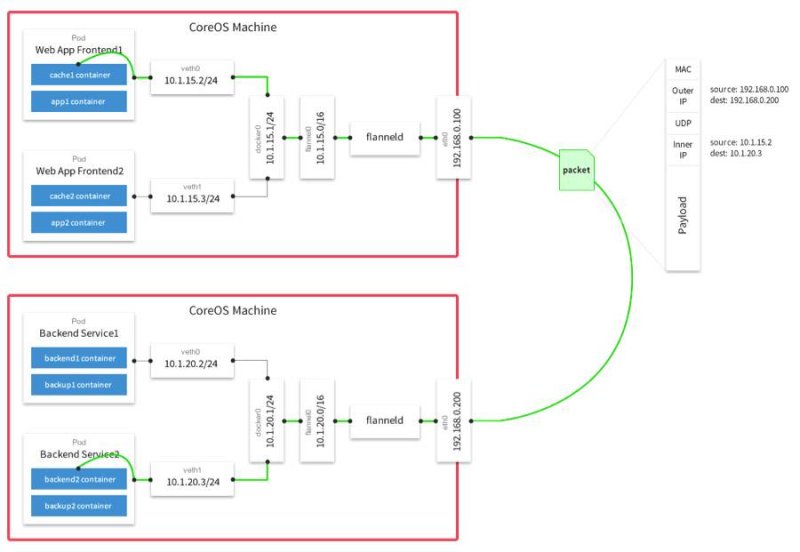
* 使用新建的bri0网桥代替缺省的docker0网桥
* bri0网桥与缺省的docker0网桥的区别：bri0和主机eth0之间是veth pair



#### Flannel(Flannel + UDP 或者 Flannel + VxLAN)

http://cdn.img2.a-site.cn/pic.php?url=aHR0cDovL21tYml6LnFwaWMuY24vbW1iaXovUVAwQVk3dGRKblV4eFJNWjRRcDl0b21GaFFRMDNYVUUzS1BRMUx0VnE3b3JvTU1BSGxSMXo3ZXhLdUtUaWJjYmVpYzdBaWFWRDNNTU1KUW1uTWlhS2t5WXlBLzA/d3hfZm10PXBuZw==  
  
Flannel实现的容器的跨主机通信通过如下过程实现：

* 每个主机上安装并运行etcd和flannel；
* 在etcd中规划配置所有主机的docker0子网范围；
* 每个主机上的flanneld根据etcd中的配置，为本主机的docker0分配子网，保证所有主机上的docker0网段不重复，并将结果（即本主机上的docker0子网信息和本主机IP的对应关系）存入etcd库中，这样etcd库中就保存了所有主机上的docker子网信息和本主机IP的对应关系；
* 当需要与其他主机上的容器进行通信时，查找etcd数据库，找到目的容器的子网所对应的outip（目的宿主机的IP）；
* 将原始数据包封装在VXLAN或UDP数据包中，IP层以outip为目的IP进行封装；
* 由于目的IP是宿主机IP，因此路由是可达的；
* VXLAN或UDP数据包到达目的宿主机解封装，解出原始数据包，最终到达目的容器。

Flannel模式如下图所示：   
  
  
演示：

#/opt/bin/etcdctl get /coreos.com/network/config  
#/opt/bin/etcdctl ls /coreos.com/network/subnets  
#/opt/bin/etcdctl get /coreos.com/network/subnets/172.16.49.0-24