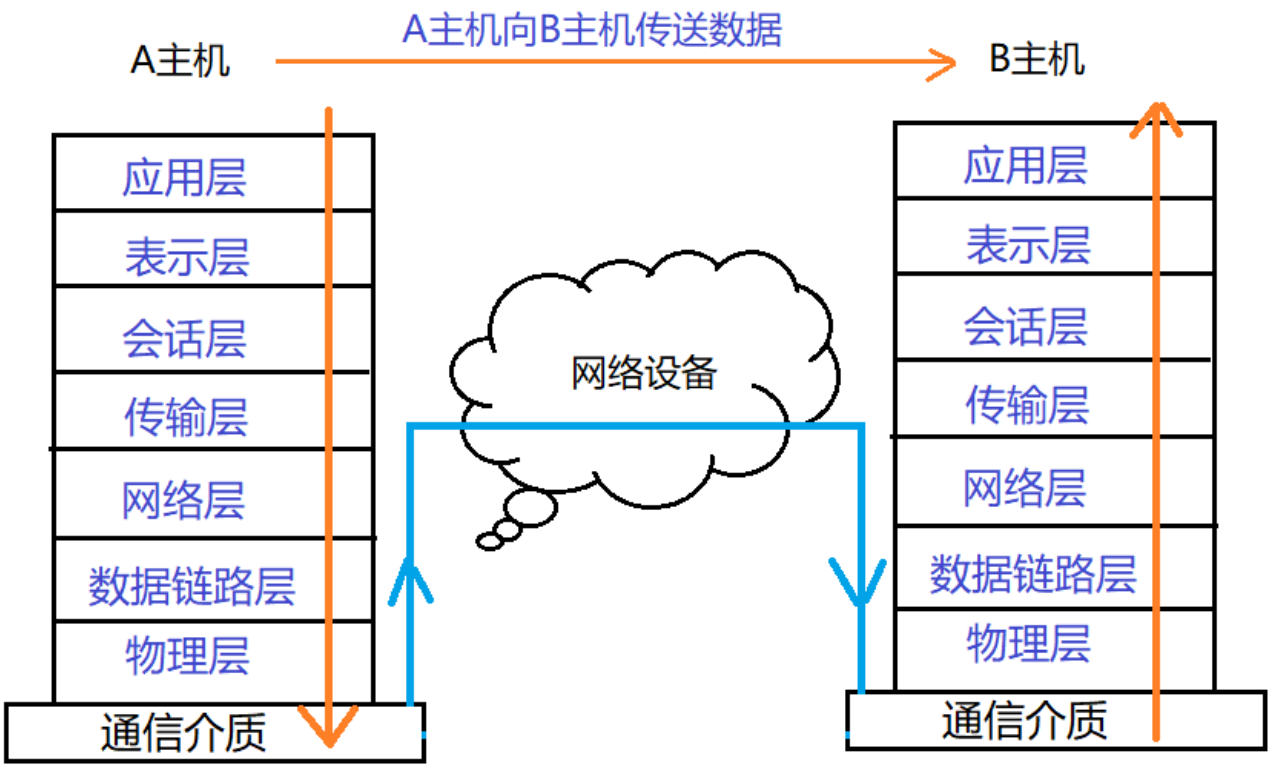
课程目标

- 了解OSI七层模型分层结构
- 了解TCP/IP协议簇四层模型分层结构
- **能够说出TCP/IP协议簇中运输层、网络层和数据链路层常见的相关协议**
- 能够说出TCP/IP的三次握手四次断开过程
- 了解Vmware的三种网络模式
- 能够使用客户端工具连接虚拟机
- 掌握主机名、DNS和静态IP的配置
- 能够使用相关命令查看和配置主机网络信息，如ifconfig/ip addr/route等

思考：

数据在两台计算机之间是如何传输的？

数据传输过程：



一、OSI七层模型

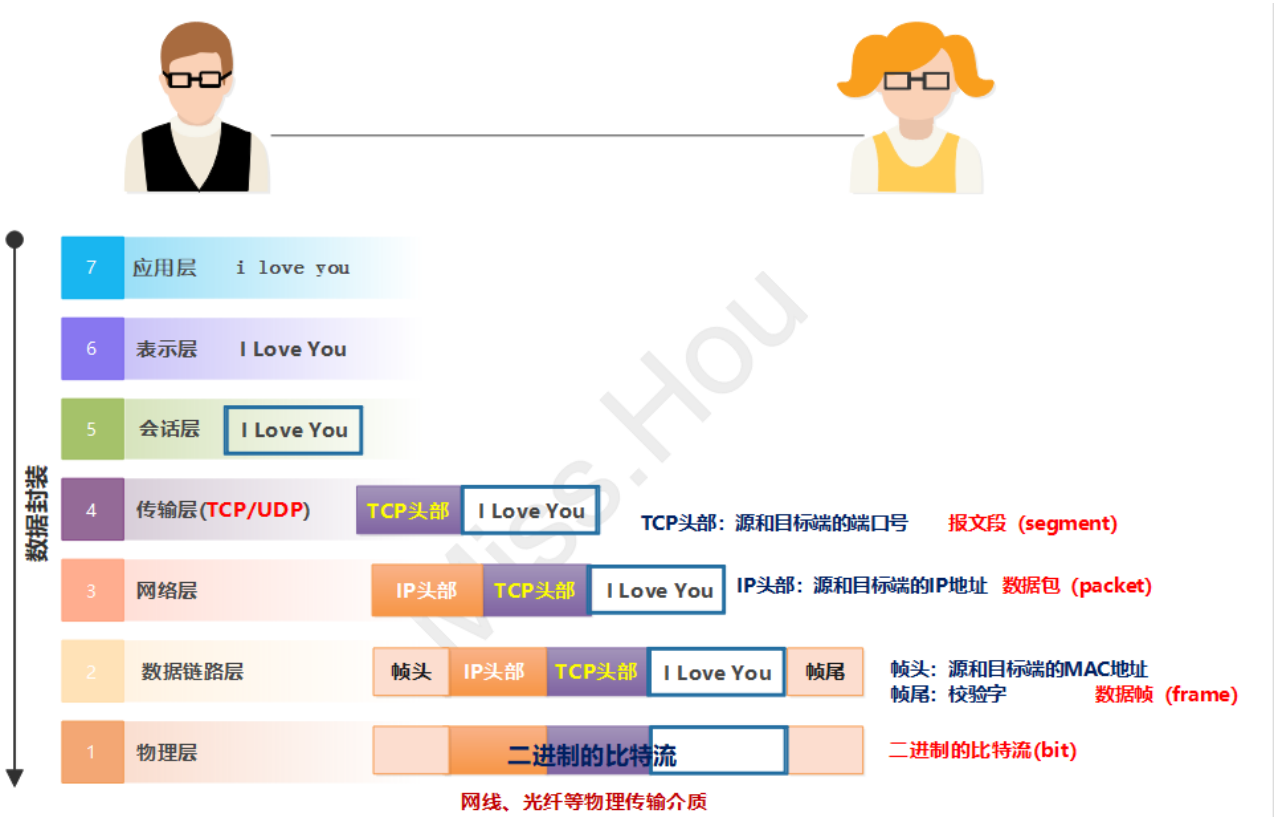
1. 什么是OSI模型

OSI:

- 开放系统互连参考模型,是国际标准化组织(ISO)和国际电报电话咨询委员会(CCITT)联合制定的开放系统互连参考模型。
- **目的**:为开放式互连信息系统提供了一种功能结构的框架和参考。

- 这里所说的开放系统，实质上指的是遵循OSI参考模型和相关协议能够实现互连的具有各种应用目的的计算机系统。
- OSI采用了分层的结构化技术，共分七层：

  物理层、数据链路层、网络层、传输层、会话层、表示层、应用层



2. OSI的七层介绍

2.1 应用层

- 应用层是计算机用户，以及各种应用程序和网络之间的接口，其功能是直接向用户提供服务，完成用户希望在网络上完成的各种工作。
- 应用层为用户提供的服务和协议：文件传输服务（FTP）、远程登录服务（ssh）、网络管理服等。
- 上述的各种网络服务由该层的不同应用协议和程序完成。
- 应用层的主要功能如下：
  - 用户接口：应用层是用户与网络，以及应用程序与网络间的直接接口，使得用户能够与网络进行交互式联系。
  - 实现各种服务：该层具有的各种应用程序可以完成和实现用户请求的各种服务。

2.2 表示层

- 表示层是**对来自应用层的命令和数据进行解释，对各种语法赋予相应的含义，并按照一定的格式传送给会话层。**
- 其主要功能是**处理用户信息的表示问题**，如编码、数据格式转换和加密解密等。
- 表示层的具体功能如下：
  - 数据格式处理：协商和建立数据交换的格式，解决各应用程序之间在数据格式表示上的差异。

    - 数据的编码：处理字符集和数字的转换。
    - 压缩和解压缩：为了减少数据的传输量，这一层还负责数据的压缩与解压缩。
    - 数据的加密和解密：可以提高网络的安全性。

2.3 会话层

- 会话层是用户应用程序和网络之间的接口，主要任务是：组织和协调两个会话进程之间的通信，并对数据交换进行管理。
- 当建立会话时，用户必须提供他们想要连接的远程地址。

2.4 传输层

- OSI上3层：应用层、表示层、会话层的主要任务是数据处理——**资源子网**
- OSI下3层：网络层、数据链路层、物理层的主要任务是数据通讯——**通讯子网**
- 传输层是OSI模型的第4层，它是通信子网和资源子网的接口和桥梁，起到承上启下的作用
- 传输层的主要任务是：**向用户提供可靠的端到端的差错和流量控制，保证报文的正确传输**

报文：报文(**message**)是网络中交换与传输的[1]

报文段:组成报文的每个分组。我们将运输层分组称为报文段(**segment**)

2.5 网络层

- 主要任务是：**数据链路层的数据在这一层被转换为**[2]**，然后通过路径选择、分段组合、顺序、进/出路由等控制，将信息从一个网络设备传送到另一个网络设备。**
- 一般情况下，数据链路层是解决**同一网络**(局域网)内节点之间的通信，而网络层主要解决**不同子网**间的通信。

2.6 数据链路层

在计算机网络中由于各种干扰的存在，物理链路是不可靠的。因此，这一层的主要功能是:

- 在物理层提供的**比特流**的基础上，通过差错控制、流量控制方法，使有差错的物理线路变为无差错的数据链路，即向网络层提供可靠的通过物理介质传输数据的方法。
- 具体工作是：接收来自物理层的位流（比特流）形式的数据，通过差错控制等方法传到网络层；同样，也将来自上层的数据，封装成[3]转发到物理层；并且，还负责处理接收端发回的确认帧的信息，以便提供可靠的数据传输。

帧：帧(**frame**)是数据链路层的传输单元。将上层传入的数据添加一个头部和尾部，组成了帧.

2.7 物理层

- 主要功能是：利用传输介质为数据链路层提供物理连接，实现比特流的透明传输。尽可能屏蔽掉具体传输介质和物理设备的差异。

3. 总结

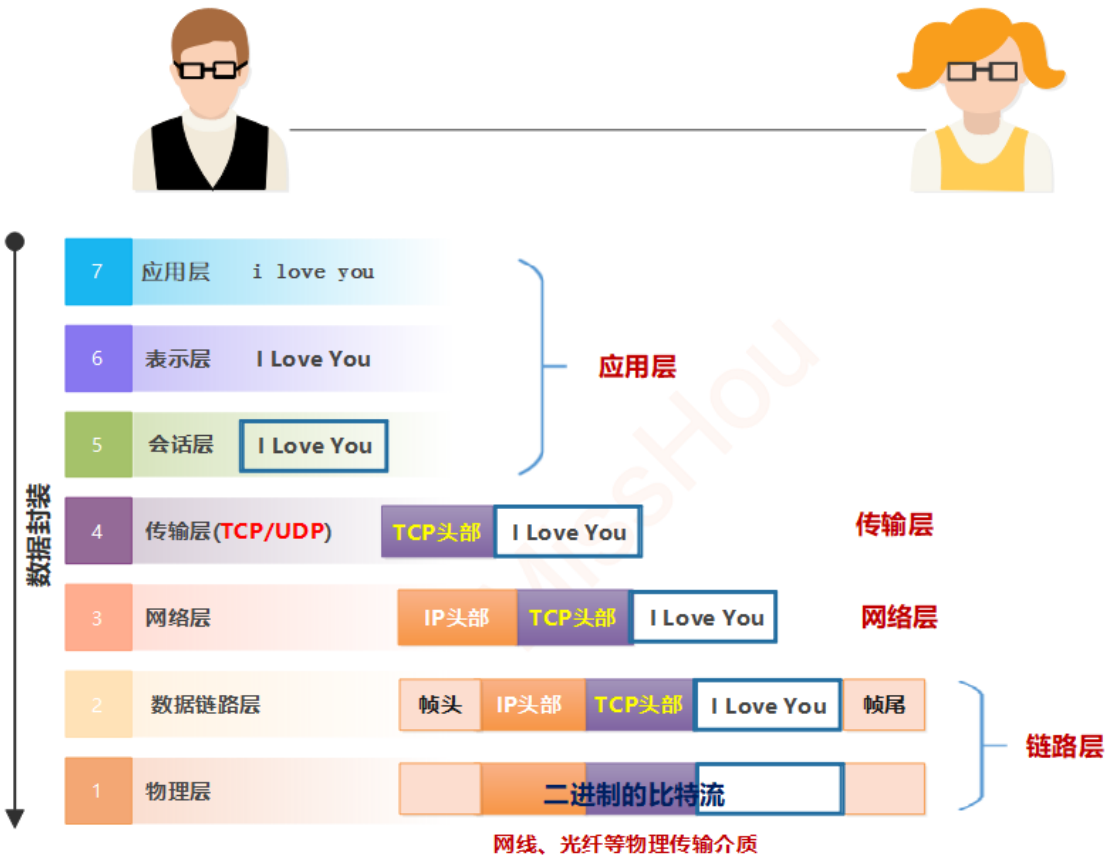
- 在7层模型中，每一层都提供一个特殊的网络功能。
- 从网络功能的角度观察：
    - 物理层、数据链路层、网络层：主要提供**数据传输和交换功能**，即节点到节点之间通信为主；
    - 传输层（第4层）：作为上下两部分的桥梁，是整个网络体系结构中最关键的部分；
    - 会话层、表示层和应用层：以提供**用户与应用程序之间的信息和数据处理**功能为主；

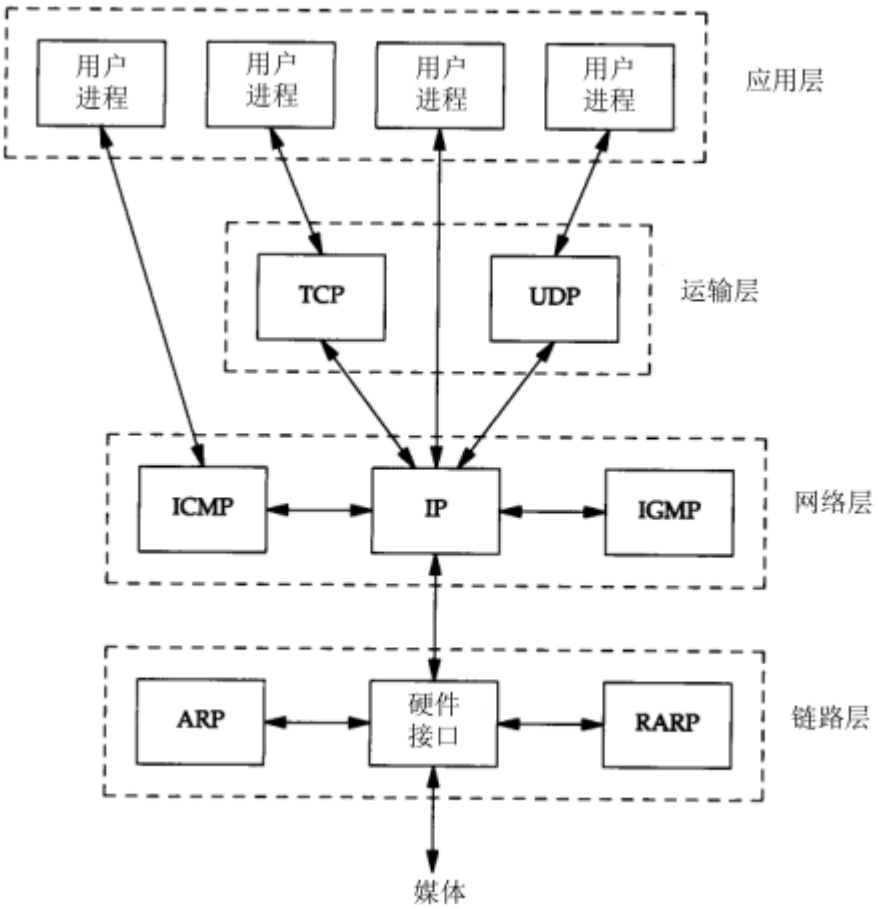
二、TCP/IP协议模型

1. 什么是TCP/IP模型

- **TCP/IP协议模型**（Transmission Control Protocol/Internet Protocol），包含了一系列构成互联网基础的网络协议，是Internet的核心协议，通过20多年的发展已日渐成熟，并被广泛应用于**局域网和广域网**中，目前已成为一种**国际标准。**
- TCP/IP协议簇是一组不同层次上的**多个协议**的组合，该协议采用了4层的层级结构，每一层都呼叫它的下一层所提供的协议来完成自己的需求，与OSI的七层模型相对应。
- 尽管通常称该协议族为TCP/IP，但TCP和IP只是其中的两种协议而已（该协议族的另一个名字是Internet协议族(Internet Protocol Suite)）

2. TCP/IP的分层结构





2.1 链路层

OSI的物理层和数据链路层

- **ARP**（地址解析协议**IP-MAC**）和**RARP**（逆地址解析协议**MAC-IP**）是某些网络接口（如以太网）使用的特殊协议，用来转换IP层和网络接口层使用的地址。

2.2 网络层

- 也称作互联网层或网际层，处理分组在网络中的活动，例如分组的选路。
- 在TCP/IP协议族中，网络层协议包括IP协议（网际协议），ICMP协议（Internet互联网控制报文协议），以及IGMP协议（Internet组管理协议）。
    - **IP**是一种网络层协议，提供的是一种不可靠的服务，它只是尽可能快地把分组从源结点送到目的结点，但是并不提供任何可靠性保证。同时被TCP和UDP使用。
    - TCP和UDP的每组数据都通过端系统和每个中间路由器中的IP层在互联网中进行传输。
    - **ICMP**是IP协议的附属协议。IP层用它来与其他主机或路由器**交换错误报文和其他重要信息**。它主要是用来提供有关通向目的地址的路径信息。Ping和Traceroute工具，它们都使用了ICMP协议。
    - **IGMP**是Internet组管理协议。它用来把一个UDP数据报多播到多个主机。该协议运行在主机和组播路由器之间。

2.3 运输层

主要为两台主机上的应用程序提供端到端的通信。在TCP/IP协议族中，有两个互不相同的传输协议：

TCP（传输控制协议）和UDP（用户数据报协议） **TCP协议：**为两台主机提供高可靠性的数据通信。TCP是面向连接的通信协议，通过**三次握手**建立连接，通讯完成时要断开连接，由于TCP是面向连接的所以只能用于端到端的通讯。TCP提供的是一种可靠的数据流服务，采用**"带重传的肯定确认"**技术来实现传输的可靠性。也就是TCP数据包中包括序号（seq）和确认（ack），所以未按照顺序收到的包可以被排序，而损坏的包可以被重传。 **UDP协议：**则为应用层提供一种非常简单的服务。它是面向无连接的通讯协议，UDP数据包括目的端口号和源端口号信息，由于通讯不需要连接，所以可以实现广播发送。 UDP通讯时不需要接收方确认，不保证该数据报能到达另一端，属于不可靠的传输，可能会出现丢包现象。UDP与TCP位于同一层，但它不管数据包的顺序、错误或重发。

2.4 应用层

OSI会话层、表示层、应用层

应用层负责处理特定的应用程序细节。

HTTP、FTP、SSH、DHCP、DNS.....

3. 数据封装过程

- **数据格式**

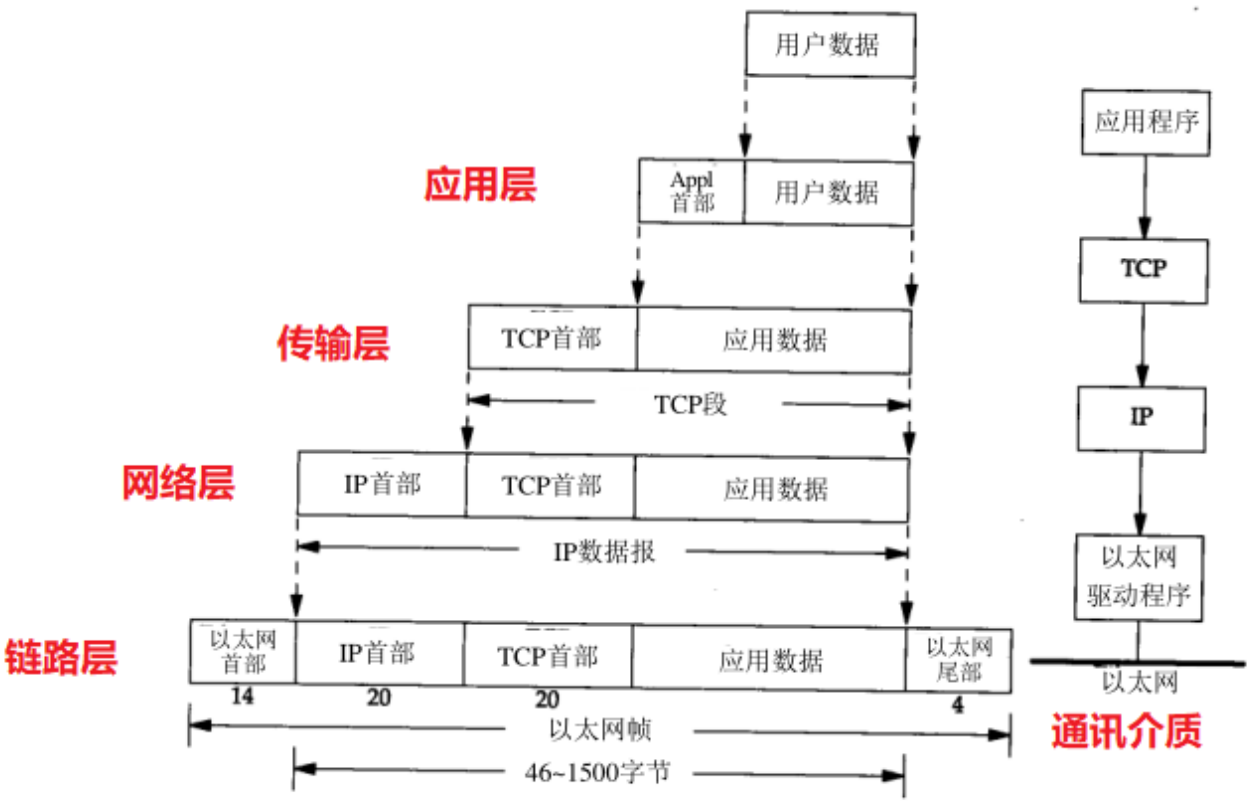
  TCP数据信息：TCP头部+实际数据 (TCP头包括源和目标主机**端口号**、顺序号、确认号、校验字等)

  IP数据包：IP头部+TCP数据信息（IP头包括源和目标主机**IP地址**、类型、生存期等）

  数据帧：帧头+IP数据包+帧尾 （帧头包括源和目标主机**MAC初步地址**及类型，帧尾是校验字）

- **数据的封装与解封装：** 封装：数据要通过网络进行传输，要从高层一层一层的向下传送，如果一个主机要传送数据到别的主机，先把数据装到一个特殊协议报头中，这个过程叫-----封装。 解封装：上述的逆向过程

  当数据以TCP/IP协议传输时的封装与街封装过程如下图：



数据进入协议栈时的封装过程

三、TCP/IP三次握手四次断开

1. 了解相关名词

序列号：Seq序号，占32位，用来标识从TCP源端向目的端发送的字节流，发起方发送数据时对此进行标记。
确认序号：Ack序号，占32位，只有ACK标志位为1时，确认序号字段才有效，Ack=Seq+1。

常见的标志位：

ACK：确认序号有效。
SYN：发起一个新连接。
FIN：释放一个连接。

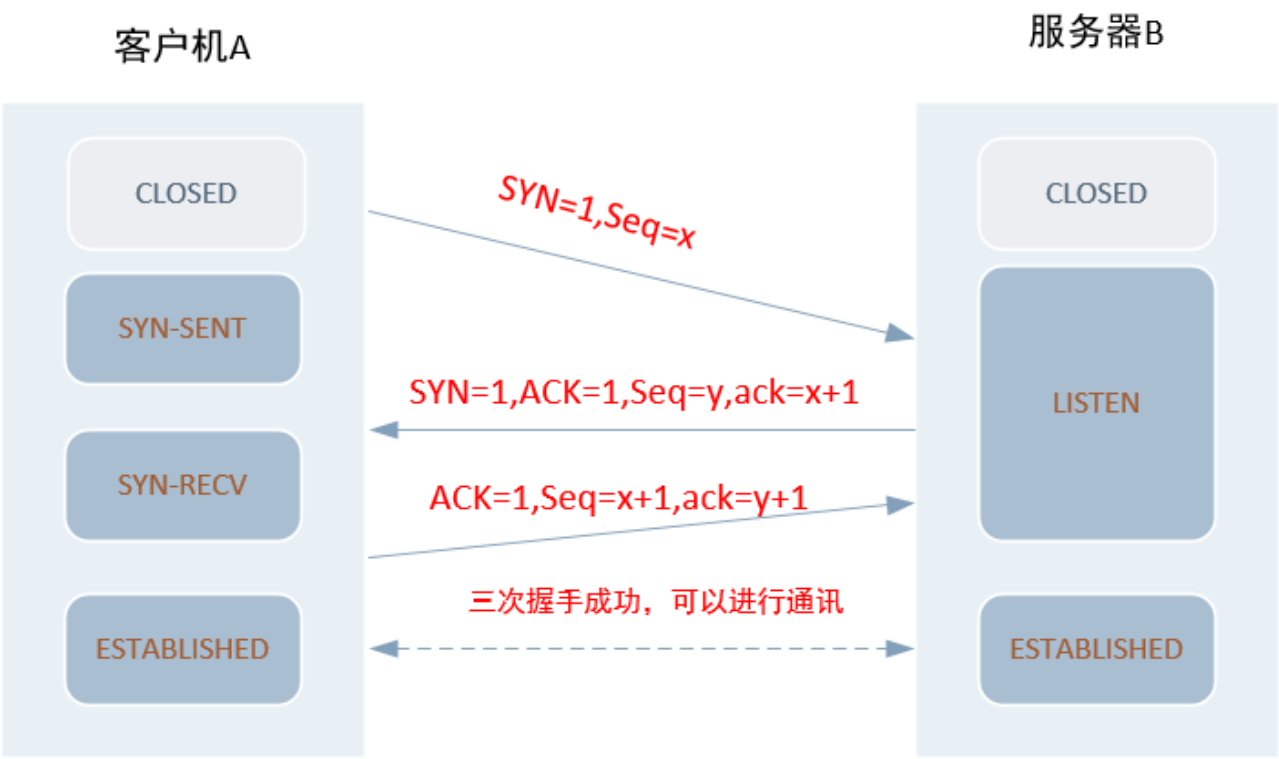
2. 了解netstat中的网络状态

CLOSED　　初始（无连接）状态。
LISTEN　　侦听状态，等待远程机器的连接请求。
SYN_SEND
在TCP三次握手中，主动连接端发送了SYN包后，进入SYN_SEND状态，等待对方的ACK包。
SYN_RECV
在TCP三次握手中，主动连接端收到ACK包后，进入SYN_RECV状态。
ESTABLISHED
完成TCP三次握手后，主动连接端进入ESTABLISHED状态。此时，TCP连接已经建立，可以进行通信。

FIN_WAIT_1　　在TCP四次断开时，主动关闭端发送FIN包后，进入FIN_WAIT_1状态。
FIN_WAIT_2　　在TCP四次断开时，主动关闭端收到ACK包后，进入FIN_WAIT_2状态。

```
TIME_WAIT       在TCP四次断开时，主动关闭端发送了ACK包之后，进入TIME_WAIT状态。
CLOSE_WAIT      在TCP四次断开时，被动关闭端收到FIN包后，进入CLOSE_WAIT状态。
LAST_ACK        在TCP四次断开时，被动关闭端发送FIN包后，进入LAST_ACK状态，等待对方的ACK包。
```

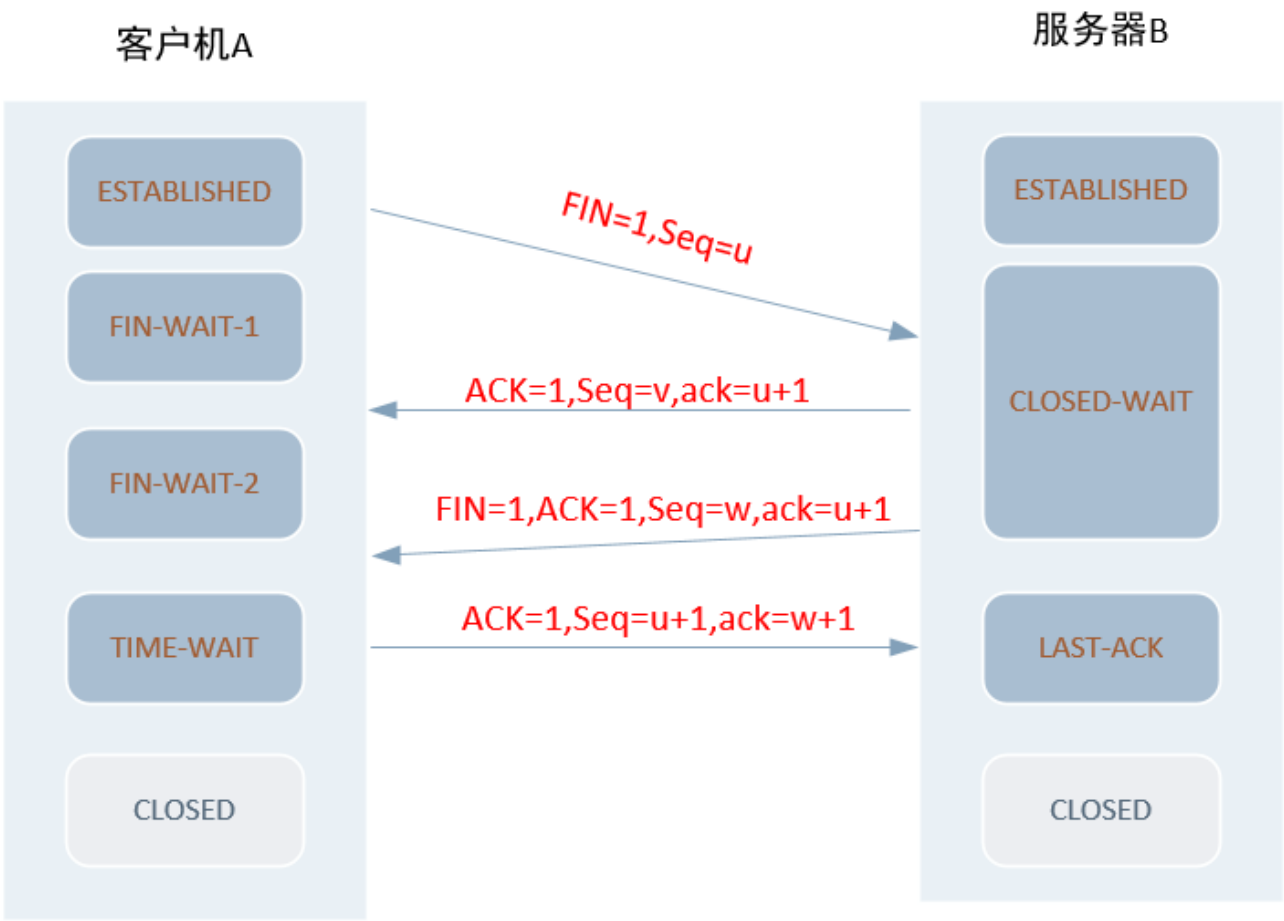
3. TCP/IP三次握手



TCP三次握手的过程如下：

1. 客户机A端（主动连接端）发送一个SYN包给服务器B端（被动连接端）；
2. 服务器B端（被动连接端）收到SYN包后，发送一个带ACK和SYN标志的包给客户机A端（主动连接端）；
3. 客户机A端（主动连接端）发送一个带ACK标志的包给服务器B端（被动连接端），握手动作完成。

4. TCP/IP四次断开



TCP四次断开的过程如下：
1. 客户机A端（主动连接端）发送一个FIN包给服务器B端（被动连接端）请求断开连接；
2. 服务器B端（被动连接端）收到FIN包后，发送一个ACK包给客户机A端（主动连接端）；
3. 服务器B端（被动连接端）发送了ACK包后，再发送一个FIN包给客户机A端（主动连接端）确认断开；
4. 客户机A端（主动连接端）收到FIN包后，发送一个ACK包，当服务器B端（被动连接端）收到ACK包后，四次断开动作完成，连接断开。

四、Vmware网络模式

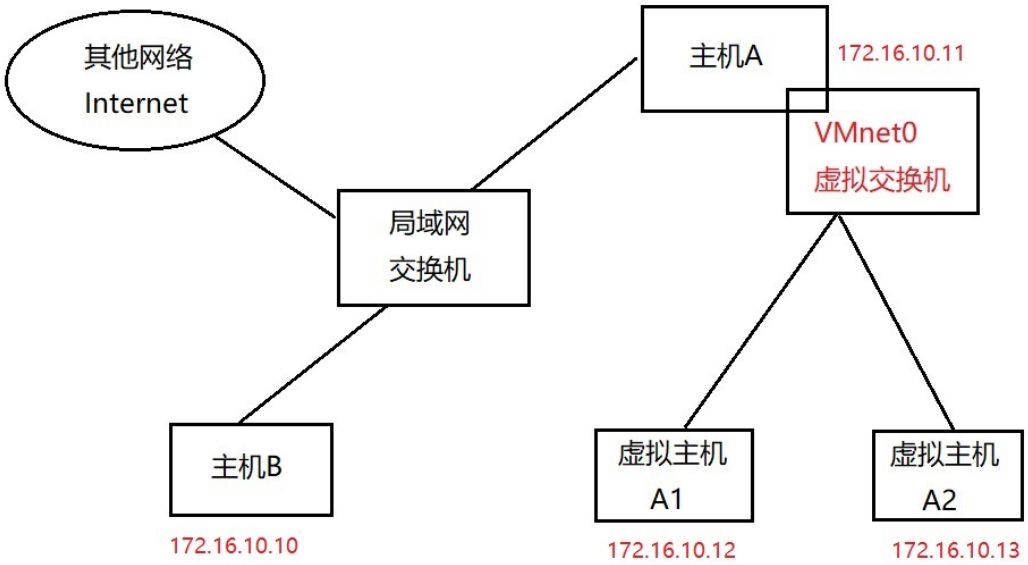
1. 虚拟设备

> VMnet0：用于虚拟**桥接网络**下的虚拟交换机
>
> VMnet1：用于虚拟**Host-Only网络**下的虚拟交换机
>
> VMnet8：用于虚拟**NAT网络**下的虚拟交换机
>
> VMware Network Adepter VMnet1：Host用于与Host-Only虚拟网络进行通信的虚拟网卡　　VMware Network Adepter VMnet8：Host用于与NAT虚拟网络进行通信的虚拟网卡

2. 三种网络模式

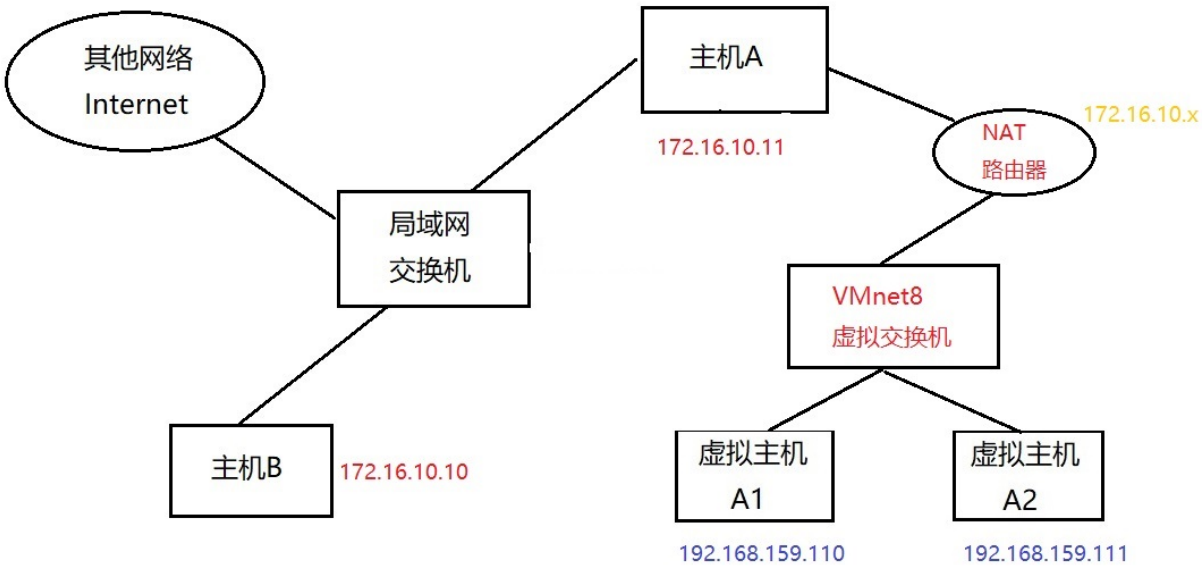
- 桥接网络

> 桥接网络是指虚拟网卡通过VMnet0虚拟交换机和本地物理网卡进行桥接，那么物理网卡和虚拟网卡就相当于处于同一个网段，虚拟交换机就相当于一台现实网络中的交换机。所以要想虚拟机也可以连接到互联网中，那么两个网卡的IP地址也要设置为同一网段。



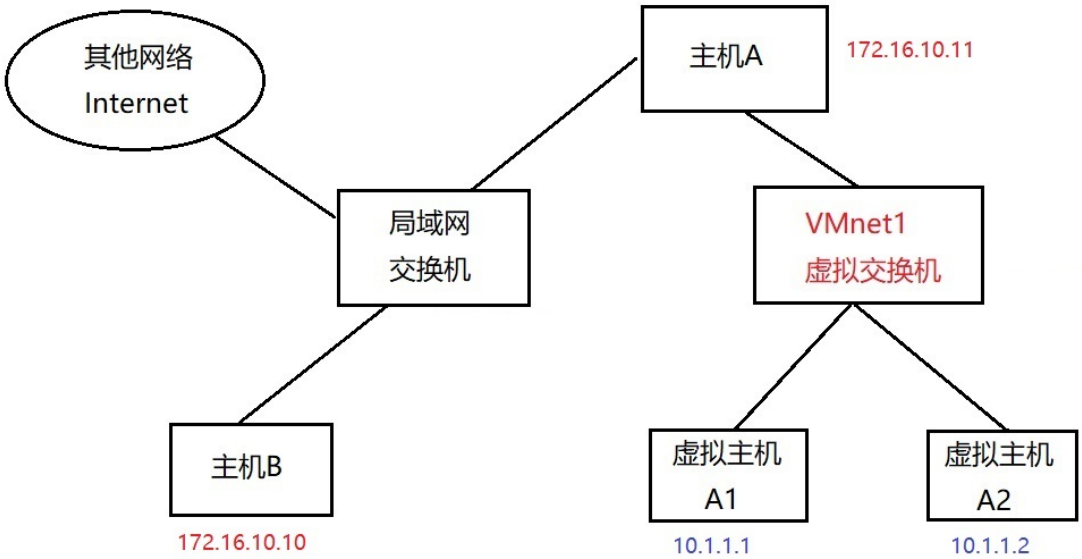
- NAT网络

> 在NAT网络中，会用到VMware Network Adepter VMnet8虚拟网卡，主机上的VMware Network Adepter VMnet8虚拟网卡被直接连接到VMnet8虚拟交换机上与虚拟网卡进行通信。VMware Network Adepter VMnet8虚拟网卡的作用仅限于和VMnet8网段进行通信，它不给VMnet8网段提供路由功能，所以虚拟机虚拟一个NAT服务器，使虚拟网卡可以连 接到Internet。 VMware Network Adepter VMnet8虚拟网卡的IP地址是在安装VMware时由系统指定生成的，我们尽量不要修改这个数值，否则可能会使主机和虚拟机无法通信。



- Host-Only模式

> 在Host-Only模式下，虚拟网络是一个全封闭的网络，它唯一能够访问的就是物理真机。其实Host-Only网络和NAT网络很相似，不同的地方就是Host-Only网络没有NAT服务，所以虚拟网络不能连接到Internet。主机和虚拟机之间的通信是通过VMware Network Adepter VMnet1虚拟网卡来实现的。



五、主机网络配置

1. 常见的网络接口

接口	描述	备注
eth0	以太网接口	eth0,eth1,ethN
wlan0	无线接口	
enp3s0/ens33	以太网接口	Centos7+
lo	本地回环接口	127.0.0.1(默认), 127.x.x.x
virbr0	桥接接口(虚拟交换机)	
br0	桥接接口(虚拟交换机)	
vnet0	KVM虚拟机网卡接口	

2. 查看网络信息

```
查看IP、掩码、MAC
[root@node1 ~]# ip addr
[root@node1 ~]# ip a
只显示eth0的信息
[root@node1 ~]# ip addr show eth0
查看本机路由表信息(默认网关，默认路由)
[root@node1 ~]# ip route
```

```
default via 10.1.1.254 dev eth0
查看DNS
[root@node1 ~]# cat /etc/resolv.conf
nameserver 119.29.29.29

ifconfig命令:
1. 给网卡配置临时子接口
# ifconfig eth0
# ifconfig -a
# ifconfig eth0:1 192.168.0.1 netmask 255.255.255.0
注意: 重启网络|系统失效

2. 永久生效需要创建子配置文件
# cp ifcfg-eth0 ifcfg-eth0:1
# pwd
/etc/sysconfig/network-scripts
[root@node1 network-scripts]# cat ifcfg-eth0:1
DEVICE=eth0:1
TYPE=Ethernet
ONBOOT=yes
BOOTPROTO=none
IPADDR=192.168.0.1
NETMASK=255.255.255.0

重启网络
# service network restart

3. 其他命令
ifup eth0
ifdown eth1
ifconfig eth0 down/up
```

环境准备要求:

1. 还原Centos6.9和Centos7.5系统
2. 以Centos6.9系统为模板克隆2台虚拟机
3. 拿一台Centos6.9系统，增加一个网卡，网络模式为仅主机模式
4. 分别配置Centos6.9系统的网络
    - 两张网卡的主机，分别配置NAT和仅主机模式IP
    - 其他两台主机，只配置仅主机模式的IP(一张网卡)

3. 修改网络信息

3.1 配置静态IP

方法1:

```
[root@node1 ~]# setup
```

方法2:

```
修改网卡配置文件
[root@node1 ~]# cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
TYPE=Ethernet
ONBOOT=yes
BOOTPROTO=none
IPADDR=10.1.1.1
NETMASK=255.255.255.0

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
DEVICE=eth0                设备名
TYPE=Ethernet              以太网
BOOTPROTO=none             IP地址获取方式，静态: static,none  动态:dhcp,dynamic
ONBOOT=yes                 重启网卡是否激活该网卡
BROADCAST=192.168.2.255     广播地址
HWADDR=00:E0:4C:41:95:DB    MAC地址
NM_CONTROLLED=yes           是否接受NetworkManager管理
IPADDR=192.168.2.253        IP地址
PREFIX=24                  子网掩码 NETMASK=255.255.255.0
NETWORK=192.168.2.0         网络地址
GATEWAY=192.168.2.254        默认网关
DNS1=202.106.0.20           DNS服务器
DNS2=8.8.8.8                DNS服务器备

+++++++++++++++++动态获取IP(dhcp)+++++++++++++++
DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes
```

3.2 修改主机名

3.3 配置DNS

```
[root@node2 ~]# cat /etc/resolv.conf
nameserver DNS服务器
nameserver 114.114.114.114
nameserver 8.8.8.8
nameserver 192.168.159.2

直接修改网卡的配置文件ifcfg-eth0:
....
DNS1=202.106.0.20           DNS服务器
DNS2=8.8.8.8                DNS服务器备
```

4. 关闭防火墙和selinux

```
Centos6.5:
临时关闭:
[root@node2 ~]# service iptables stop
iptables: Setting chains to policy ACCEPT: filter          [  OK  ]
iptables: Flushing firewall rules:                         [  OK  ]
iptables: Unloading modules:                               [  OK  ]
```

```
[root@node2 ~]# service iptables status
iptables: Firewall is not running.
[root@node2 ~]#
开机自动关闭:
[root@node2 ~]# chkconfig --list|grep iptables
iptables        0:off   1:off   2:on    3:on    4:on    5:on    6:off
[root@node2 ~]# chkconfig iptables off

关闭selinux:
[root@node2 ~]# getenforce
Enforcing
[root@node2 ~]# setenforce
usage:  setenforce [ Enforcing | Permissive | 1 | 0 ]
[root@node2 ~]# setenforce 0            临时变成警告模式
[root@node2 ~]# getenforce
Permissive
[root@node2 ~]# cat /etc/selinux/config
...
SELINUX=disabled                //关闭selinux，下次开机生效
...
```

5. 其他工具

```
lspci：显示系统中所有PCI总线设备或连接到该总线上的所有设备的工具
//查看当前主机的所有网卡（包括已经驱动了和没有驱动）
[root@misshou ~]# lspci |grep -i eth
00:03.0 Ethernet controller: Red Hat, Inc Virtio network device

//查看物理连接状态（网线是否ok）
[root@misshou ~]# ethtool eth0
Settings for eth0:
        Link detected: yes

[root@node1 ~]# mii-tool eth0
eth0: negotiated 100baseTx-FD, link ok
```

总结:

网络配置：静态IP

主机名配置：完全规范主机名 server server.heima.cc

IP地址和主机名一一绑定写到host文件中

关闭防火墙和selinux

---

1. 数据单元是网络信息传输的基本单位。一般网络连接不允许传送任意大小的数据包，而是采用分组技术将一个数据分成若干个很小的数据包，并给每个小数据包加上一些关于此数据包的属性信息.↩

2. 包(Packet)是TCP/IP协议通信传输中的数据单位，一般也称"数据包"。↩

3. 帧是数据链路层的传输单元。它将上层传入的数据添加一个头部和尾部，组成了帧.↩