UC Day14

预习课

预习内容

网络基础

网络基础

什么是计算机网络

知识讲解

- 计算机网络，是指将地理位置不同的具有独立功能的多台计算机及其外部设备，通过通信线路连接起来，在网络操作系统、网络管理软件及网络通信协议的管理和协调下，实现资源共享和信息传递的计算机系统

什么是网络协议

知识讲解

- 网络协议是一种特殊的软件，是计算机网络实现其功能的最基本的机制。网络协议的本质就是规则，即各种硬件和软件必须遵循的共同守则。网络协议并不是一套单独的软件，它融合于其它所有的软件甚至硬件系统中，因此可以说协议在网络中无所不在

什么是协议栈

- 为了减少网络设计的复杂性，绝大多数网络采用分层设计的方法。所谓分层设计，就是按照信息的流动过程将网络的整体功能分解为一个个的功能层，不同机器上的同等功能层之间采用相同的协议，同一机器上的相邻功能层之间通过接口进行信息传递。各层的协议和接口统称为协议栈

知识讲解

什么是协议栈

- 描述计算机网络各协议层的一般方法是采用国际标准化组织(International Standardization Organization, ISO)的计算机通信开放系统互连(Open System Interconnection, OSI)模型，简称ISO/OSI网络协议模型

上层协议定义网络数据的格式及各种网络应用	应用层	为用户的应用程序提供各种网络服务	http/ftp/telnet
	表示层	将不同数据格式转换为一种通用格式	ascii/jpeg/mpeg
	会话层	建立、管理和终止通信主机间的对话	安排访问次序
底层协议定义数据如何传输到目的地	传输层	提供端到端的流量控制维持可靠传输	TCP/UDP
	网络层	路径选择、路由、IP寻址、建立连接	IP/SPX、路由器
	数据链路层	物理寻址、网络拓扑结构、错误检测	MAC、网桥
	物理层	高低电平、速率与距离、物理连接器	HUB/中继器 线缆

TCP/IP

- TCP/IP不是个单一的网络协议，而是由一组具有层次关系的网络协议组成的协议家族，简称TCP/IP协议族
  - TCP：传输控制协议，面向连接，可靠的全双工的字节流
  - UDP：用户数据报协议，无连接，不如TCP可靠但速度快
  - ICMP：网际控制消息协议，处理路由器和主机间的错误和控制消息
  - IGMP：网际组管理协议，用于多播
  - IPv4：网际协议版本4，使用32位地址，为TCP、UDP、ICMP和IGMP提供递送分组服务
  - IPv6：网际协议版本6，使用128位地址，为TCP、UDP和ICMPv6提供递送分组服务

知识讲解

TCP/IP

- TCP/IP不是个单一的网络协议，而是由一组具有层次关系的网络协议组成的协议家族，简称TCP/IP协议族
  - ARP：地址解析协议，把IPv4地址映射到硬件地址
  - RARP：逆地址解析协议，把硬件地址映射到IPv4地址
  - ICMPv6：网际控制消息协议版本6，综合了ICMP、IGMP和ARP的功能
  - BPF：BSD分组过滤器，为应用程序提供访问数据链路层的接口，由源自BSD的系统内核提供
  - DLPI：数据链路提供者接口，为应用程序提供访问数据链路层的接口，由源自SVR4的系统内核提供

知识讲解

TCP/IP

知识讲解

- 在ISO/OSI网络协议模型的基础上，TCP/IP协议做了部分合并和简化，同时将网络编程的接口设定在传输层与会话层之间，这样做的理由有二
  - 上三层与应用程序的业务逻辑(如数据包的组织与解析、收发的时机与次序等)密切相关，而与具体的通信细节(如收发分组、等待确认、分组排序、计算验证校验和丢包重传等)关系不大；下四层主要处理通信细节而与具体应用的业务逻辑无关
  - 上三层通常构成用户进程，而下四层通常是系统内核的一部分

TCP/IP

知识讲解


应用层
表示层
会话层
套接字接口
应用层协议
http/ftp等
传输层
XTI接口
TCP/UDP
网络层
IPv4/IPv6
数据链路层
设备驱动
和硬件
物理层

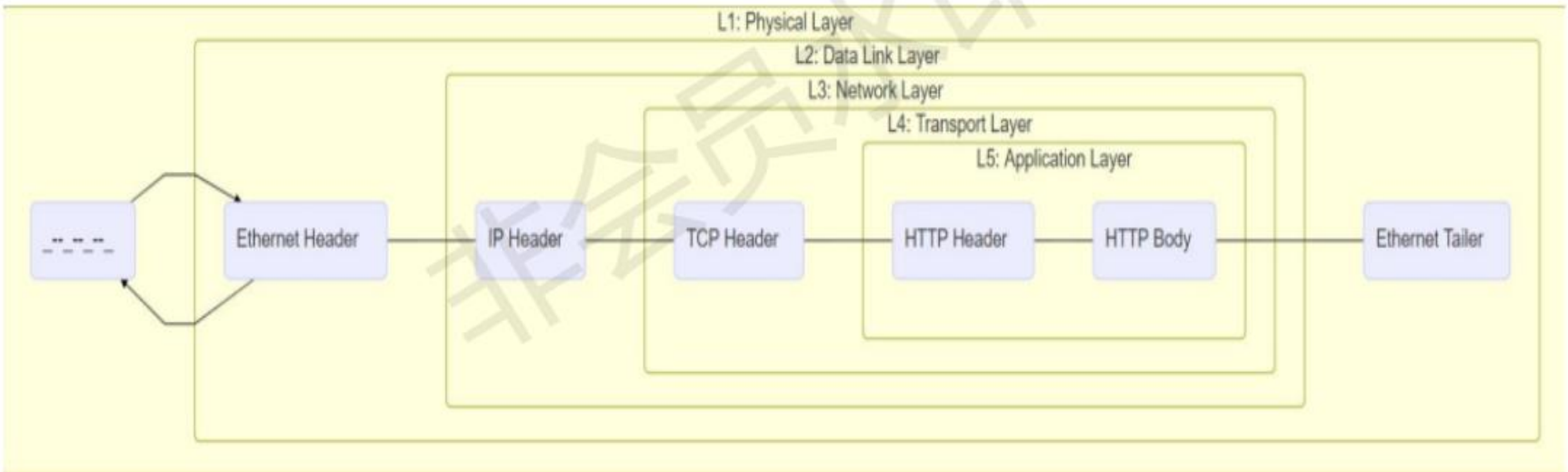
消息包和消息流

知识讲解

- 应用程序负责组织的通常都是与业务相关的数据内容，而要想把这些数据内容通过网络发送出去，就要将其自上向下地压入协议栈，每经历一个协议层，就会对数据做一层封包，每一层输出的封包都是下一层输入的内容，消息包沿着协议栈的运动形成了消息流
- 当从网络上接收数据时，过程刚好相反，消息包自下向上地流经协议栈，每经历一个协议层，就会对输入的数据解一层封包，经过层层解包以后，应用程序最终得到的将只是与业务相关的数据内容

消息包和消息流

- 数据的封装和解析过程



知识讲解

直播课见

UC

C/C++教学体系

目录

IP地址

IP地址

知识讲解

- 什么是IP地址?
  - IP地址，全称网际协议地址(Internet Protocol Address)，是IP协议提供的一种统一的地址格式，为互联网上的每个网络和每台主机分配一个逻辑地址，借以消除物理地址差异性所带来的影响
- IP地址如何表示?
  - 在计算机内部，IP地址用一个32位无符号整数表示，如：0x01020304。
  - 人们更习惯使用点分十进制字符串表示，如：1.2.3.4。字符串形式的从左到右，对应整数形式的从高字节到低字节。注意这里所说的高低指的是数位高低而非地址高低

IP地址

知识讲解

- 什么是IP地址分级？
  - A级地址：以0为首的8位网络地址+24位本地地址
  - B级地址：以10为首的16位网络地址+16位本地地址
  - C级地址：以110为首的24位网络地址+8为本地地址
  - D级地址：以1110为首的32位多播地址
  - 例如：某台计算机的IP地址：192.168.182.48，写成整数形式：11000000 10101000 10110110 00110000，C级地址，网络地址：192.168.182.0，本地地址：48

IP地址

知识讲解

- 借助子网掩码可以快速帮助我们区定IP地址的网络地址和本地地址
  - 以IP地址：192.168.182.48，子网掩码：255.255.255.0为例
  - 网络地址 = IP地址 & 子网掩码

    192.168.182.48 & 255.255.255.0 = 192.168.182.0
  - 本地地址 = IP地址 & ~子网掩码

    192.168.182.48 & 0.0.0.255 = 0.0.0.48

套接字

套接字

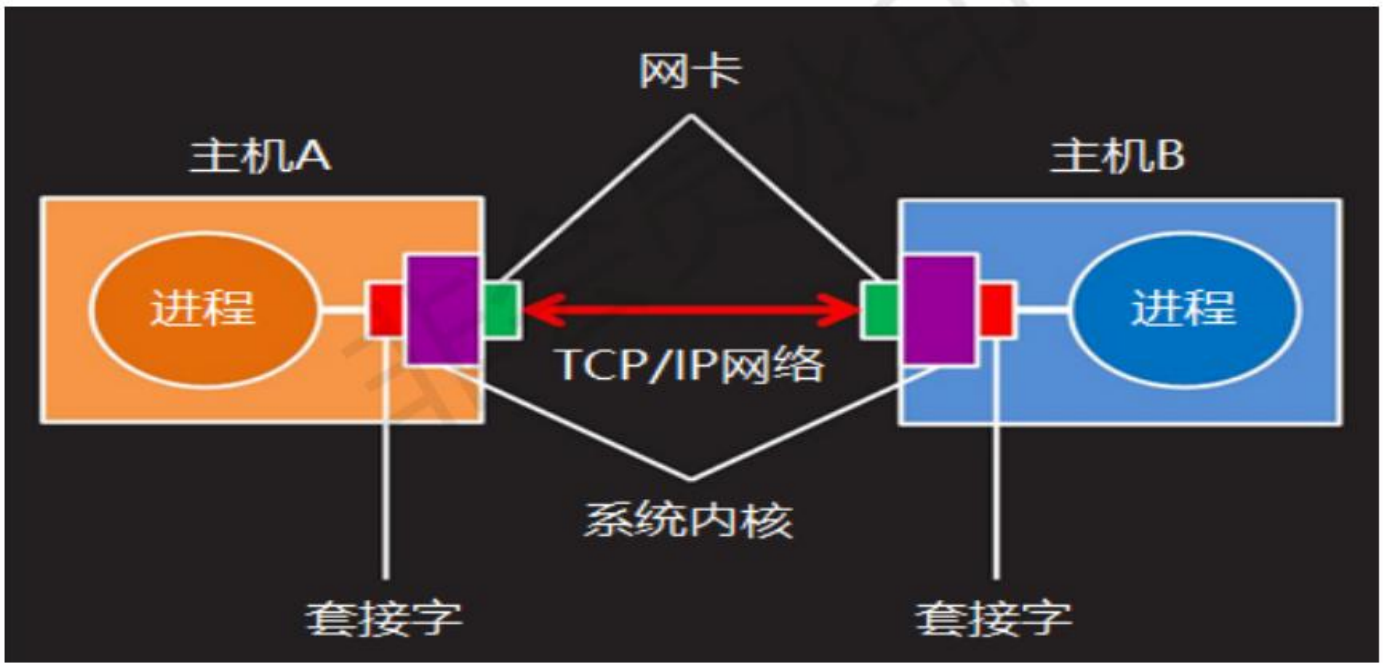
知识讲解

- 什么是套接字?
  - 套接字(socket)的本意是指电源插座，这里将其引申为一个基于TCP/IP协议可实现基本网络通信功能的逻辑对象
  - 机器与机器的通信，或者进程与进程的通信，在这里都可以被抽象地看作是套接字与套接字的通信
  - 应用程序编写者无需了解网络协议的任何细节，更无需知晓系统内核和网络设备的运作机制，只要把想发送的数据写入套接字，或从套接字中读取想接收的数据即可

套接字

知识讲解

- 什么是套接字?



套接字

知识讲解

- 什么是套接字?
  - 从这个意义上讲，套接字就相当于一个文件描述符，而网络就是一种特殊的文件，面向网络的编程与面向文件的编程已没有分别，而这恰恰是Unix系统一切皆文件思想的又一例证
  - 套接字是对ISO/OSI网络协议模型中传输层及其以下诸层的逻辑抽象，是对TCP/IP网络通信协议的高级封装，因此无论所依赖的是什么硬件，所运行的什么操作系统，所使用的是什么编程语言，只要是基于套接字构建的应用程序，只要是在互联网环境中通信，就不会存在任何障碍

套接字

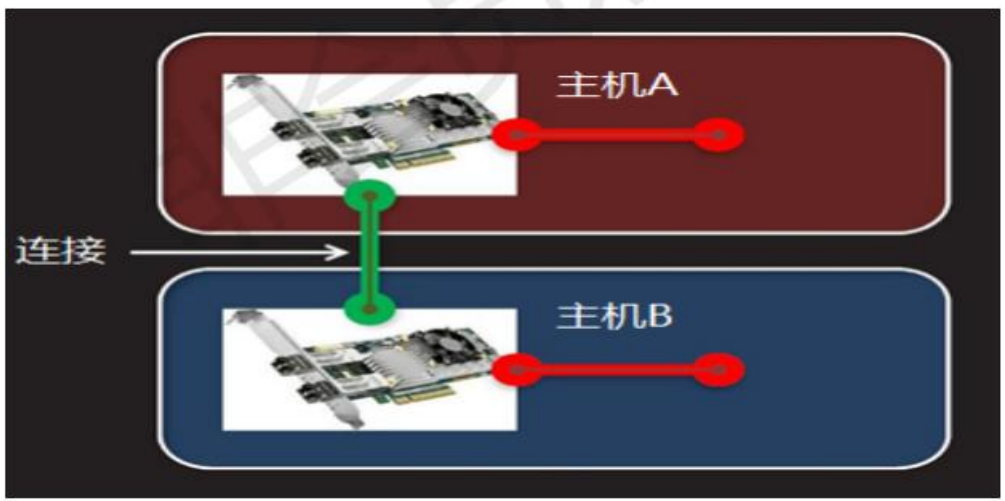
知识讲解

- 如前所述，套接字是一个提供给程序员使用的逻辑对象，它表示对ISO/OSI网络协议模型中传输层及其以下诸层的抽象。但真正发送和接收数据的毕竟是那些实实在在的物理设备。这就需要在物理设备和逻辑对象之间建立一种关联，使后续所有针对这个逻辑对象的操作，最终都能够反映到实际的物理设备上。建立这种关联关系的过程就叫做绑定

套接字

知识讲解

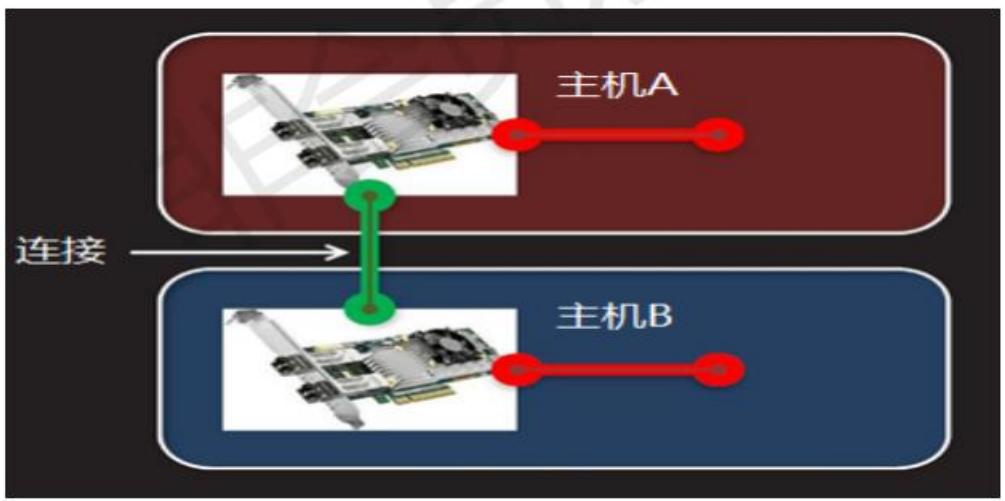
- 绑定只是把套接字对象和一个代表自己的物理设备关联起来。但为了实现通信还需要把自己的物理设备与对方的物理设备关联起来。只有这样才能建立起一种以物理设备为媒介的，跨越不同进程甚至机器的，多个套接字对象之间的联系。建立这种联系的过程就叫做连接



套接字

知识讲解

- 绑定只是把套接字对象和一个代表自己的物理设备关联起来。但为了实现通信还需要把自己的物理设备与对方的物理设备关联起来。只有这样才能建立起一种以物理设备为媒介的，跨越不同进程甚至机器的，多个套接字对象之间的联系。建立这种联系的过程就叫做连接



相关函数

相关函数

知识讲解

- #include <sys/socket.h>
- int socket(int domain, int type, int protocol);
  - 功能：创建套接字
  - 参数：domain：通信域，协议族，可取以下值：

    PF_LOCAL/PF_UNIX - 本地套接字，进程间通信

    PF_INET - 基于IPv4的网络通信

    PF_INET6 - 基于IPv6的网络通信

    PF_PACKET - 基于底层包的网络通信

相关函数

知识讲解

- #include <sys/socket.h>
- int socket(int domain, int type, int protocol);
  - 参数：type：套接字类型，可取以下值：
    SOCK_STREAM - 流式套接字，基于TCP协议
    SOCK_DGRAM - 数据报套接字，基于UDP协议
    SOCK_RAW - 原始套接字，工作在传输层以下
    protocol：特殊协议，对于流式和数据报套接字而言，只能取0
  - 返回值：成功返回表示套接字对象的文件描述符，失败返回-1。

相关函数

知识讲解

- 套接字接口库通过地址结构定位一个通信主体，可以是一个文件，可以是一台远程主机，也可以是执行者自己

  - 基本地址结构，本身没有实际意义，仅用于泛型化参数

```
struct sockaddr {
        sa_family_t sa_family;   // 地址族
        char        sa_data[14]; // 地址值
};
```

相关函数

知识讲解

- 本地地址结构，用于AF_LOCAL/AF_UNIX域的本地通信

```
struct sockaddr_un {
        sa_family_t sun_family; // 地址族(AF_LOCAL/AF_UNIX)
        char        sun_path[]; // 本地套接字文件的路径
};
```

- 网络地址结构，用于AF_INET域的IPv4网络通信

```
struct sockaddr_in {
        sa_family_t     sin_family; // 地址族(AF_INET)
        in_port_t       sin_port;   // 端口号(0~65535) - unsigned short
        struct in_addr sin_addr;    // IP地址          - unsigned int
};
```

相关函数

知识讲解

– 网络地址结构，用于AF_INET域的IPv4网络通信

```
struct in_addr {
        in_addr_t s_addr;
};

typedef uint16_t in_port_t; // 无符号16位整数
typedef uint32_t in_addr_t; // 无符号32位整数
```


相关函数

知识讲解

- 如前所述，通过IP地址可以定位网络上的一台主机，但一台主机上可能同时有多个网络应用在运行，究竟想跟哪个网络应用通信呢？这就需要靠所谓的端口号来区分，因为不同的网络应用会使用不同的端口号。用IP地址定位主机，再用端口号定位运行在这台主机上一个具体的网络应用，这样一种对通信主体的描述才是唯一确定的
- 套接字接口库中的端口号被定义为一个16位的无符号整数，其值介于0到65535，其中0到1024已被系统和一些网络服务占据，比如21端口用于ftp服务、23端口用于telnet服务、80端口用于www服务等，因此一般应用程序最好选择1024以上的端口号，以避免和这些服务冲突

相关函数

知识讲解

- #include <sys/socket.h>
- int bind(int sockfd, struct sockaddr const* addr, socklen_t addrlen);
  - 功能：将套接字和本机的地址结构绑定在一起
  - 参数：sockfd：套接字描述符。
    addr：自己的地址结构。
    addrlen：地址结构的字节数
  - 返回值：成功返回0，失败返回-1。

相关函数

知识讲解

- #include <sys/socket.h>
- int connect(int sockfd, struct sockaddr const* addr, socklen_t addrlen);
  - 功能：将套接字和对方的地址结构连接在一起
  - 参数：sockfd：套接字描述符。
    addr：对方的地址结构。
    addrlen：地址结构的字节数
  - 返回值：成功返回0，失败返回-1。

字节序转换

字节序转换

知识讲解

- 网络应用与单机应用不同，经常需要在具有不同硬件架构和操作系统的计算机之间交换数据，因此编程语言里一些多字节数据类型的字节序问题就需要特别予以关注
- 套接字接口库规定在网络传输过程中采用网络字节序，也就是大端字节序，而本机数据可能是小短字节序
  - 小端字节序：数据的低位存放在低地址
  - 大端字节序：数据的低位存放在高地址

字节序转换

知识讲解

- 转换函数
  - uint32_t htonl(uint32_t hostlong); //长整形主机字节序到网络字节序
  - uint32_t ntohl(uint32_t netllong); //长整形网络字节序到主机字节序
  - uint16_t htons(uint16_t hostshort);//短整形主机字节序到网络字节序
  - uint16_t ntohs(uint16_t netshort);//短整形网络字节序到主机字节序
  - in_addr_t inet_addr(char const* ip);
    点分十进制字符串地址 --》 网络字节序形式整数地址
  - int inet_aton(char const* ip, struct in_addr* nip);
    点分十进制字符串地址 --》 网络字节序形式整数地址
  - char* inet_ntoa(struct in_addr nip);
    网络字节序形式整数地址 --》 点分十进制字符串地址

谢谢

UC Day014

复习课

客户端和服务器

客户端和服务器

- 客户端：
  - 客户端又称为用户端，是指与服务器相对应，为客户提供本地服务的程序。在大家网上冲浪的过程中，我们就是客户，而我们所使用的用来获取网络信息的工具就是客户端，比如浏览器，比如各类手机APP，视频的播放器等。
  - 对于这一类的应用程序，需要网络中有相应的服务器和服务程序来提供相应的服务，如数据库服务，电子邮件服务等等，这样在客户机和服务器端，需要建立特定的通信连接，来保证应用程序的正常运行

知识讲解

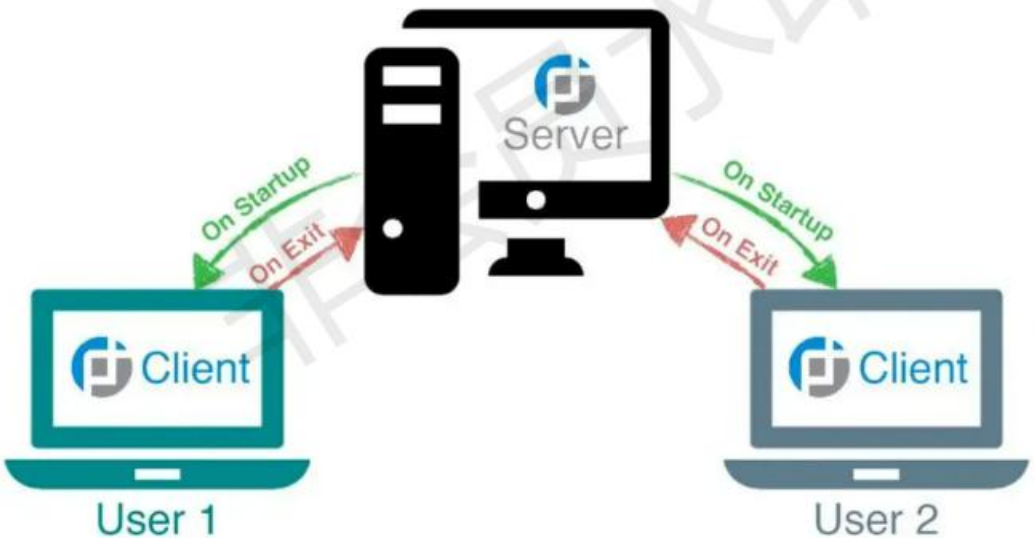
客户端和服务器

- 服务器：
  - 服务端是为客户端服务的，服务的内容诸如向客户端提供资源，保存客户端数据等。
  - 服务器上会存储大量的资源，比如音频文件，视频文件，图片，文字等内容，当客户端访问服务器时，服务器就会将相应资源传递给服务器。
  - 服务器上也会保存客户端的数据，比如我们自身的账号信息，交易记录，游戏数据等。

知识讲解

客户端和服务器

- 平时我们所说的网络通信，实际指的就是客户端和服务器之间的通信，借助响应的通信协议，客户端和服务器之间实现数据的传递和信息的交互。



知识讲解

下节课见