第二章 应用层

2.1 网络应用

End systems（运行网络应用程序的计算机系统）

End-to-end communication（communication between processes）端对端通信

通信双方：clients—server（分布式）程序有两部分组成（客户端、服务器）

Web应用：browser、web server

我们编写客户端程序Client，也得编写服务器程序server，这些程序在端系统上运行，不是在网络核心的网络设备上运行，两者间通讯必须就双方的通讯信息内容、格式和交互（协议，protocol）进行约定，通常客户向服务器发送的叫请求（request），服务器向客户返回的叫应答（response）。

目前所有的应用都是C/S（client/server）架构

在网络设备上是否需要编写程序呢？

也需要编写一些程序，这些程序通常是承担对网络设备进行管理的程序。

服务器程序应该在client运行前先运行，server运行在等待client端请求—向服务器发送命令：

命令名称，

命令处理的对象—数据

服务器等待客户端的请求，一个服务器可以处理很多客户端的请求，客户请求可能同时到达，如何处理请求？

处理请求有两种：

* 迭代处理—一个接着一个处理，在前一个处理完之前，后面的请求排队
* 并发处理—每个请求都分派专门的处理单位（进程、线程）进行处理

服务器接收请求后对请求进行解析，按照请求的要求进行相应处理，处理完后把结果重新按照协议要求打包成响应返回给客户端，客户端将返回的响应进行解析，解析后进行相应处理。数据库—使用的协议通常的是专有的

* 客户端提供界面（GUI）,程序接受用户的对GUI操作，把这些操作组装成SQL语句，发送给数据库管理系统（database engine）
* 数据库管理系统（Database Engine）把接收到的sql语句进行分析，按照SQL语句的要求进行，把进行的结果打包后返回给客户端
* 客户端接收到服务器的响应进行相应的处理

客户端少量内存泄露问题不算严重，但内存泄露对服务器而言就非常严重。

P2P（peer to peer）是变通的C/S，既有Client端，又有Server端，client向其它节点的server请求资源，本地server端将本地client端已获取的资源向其它节点的客户端提供服务，这样实现我为人人，人人为我，通常用于文件共享，有哪些文件类型： 文章、电子书、音乐、电影等，这些大多数有版权，侵权。

通信主体：进程

两个进程，可能在同一个节点，也可能在不同节点

同一节点间两个进程通信有以下机制可以实现：

IPC（Interprocess Communication）进程间通信

* 效率高
* 在OS统一管理，所有资源都可以在OS下进行寻址，一个数字
* IPC机制：
  + Pipe
  + File
  + 信号灯
  + Message Slot
  + Shared Memory
  + Mailslot

Socket（既可以用于同一节点内部，也可以实现不同节点间的通讯）

寻址方式不一样，需要指出收发双方的地址，涉及到将输出的message打包，接收时解包

进程寻址：

分两级寻址:

* 节点—IP地址
* 节点内部，采用端口号（port number）：16bit（0—65535）

1. 客户端端口号采用申请制—随机分配，需要网络通信时申请，不需要就不用，临时分配，值是不固定。使用时申请，用完后释放，释放时便于反复使用，复用。
2. 绑定制，服务器端采用的，固定在某一个端口上，可以把固定端口号公开。
3. 为什么采用两种不同方式呢？

是根据客户端和服务器端各自不同的特点所决定的。

常用的端口：80（http），443（https，使用的是SSL（secure Socket Layer协议），SSL现在不怎么安全，TLS（Transport Layer Security）。

HTTP使用的明码传输，不符合以下要求：

1. 机密性
2. 完整性
3. 防

大多数使用https协议进行传输

开放协议：

协议内容公开的，客户端和服务器可以分开开发。如Web，browser：chrome、firefox，safari等等；Web Server: Apache、Nginx等等。

私有协议：Skype、QQ、微信、Oracle数据库（oci）

传输服务要求（QoS）：

延时（时间敏感性）

吞吐量：实时要求高的原因：

实时控制（导弹控制、自动驾驶）、多媒体（视频和音频）

可靠：数据丢失

TCP/IP协议传输层提供三种服务供开发者选择：

TCP—可靠协议，1：1通信

UDP—不可靠协议—多媒体传输，实现1：多通信

STCP（UNIX网络程序设计）有多个通路时可以同时使用两个以上通路

手机有两个与网络连接的通路：WLAN，4G/5G

2.2 Web

一、节点标识：

* Mac地址（局部地址，某一局部逻辑网段）
* IP地址（全局地址，范围限制）
* 域名（[www.github.com](http://www.github.com))

域名 IP地址 mac地址



域名到IP地址转换：Domain Service（DNS）

IP地址到域名转换：Domain Service（电子交易时可以查看IP地址对应的域名）

nslookup命令行程序可以帮助查看转换

IP地址到MAC地址转换：ARP协议实现的

MAC地址到IP地址转换：RARP协议实现的

HTTP 1.0协议采用的是一个来回只传输一个html对象（一个HTML文件），传输前要进行TCP连接，传输完关闭TCP连接，重新前面的过程，这种连接是无状态的。

HTTP 1.1 持久化的HTTP(Persistent HTTP)，在1.0最后关闭连接时不操作，在同一连接持续进行传输，节省了后续的传输连接建立时间。

RTT(Round trip Time)往返时间

TCP连接的过程：

1. 客户端（Browser）发起TCP链接请求（syn=1），有请求顺序号（client seq#，随机生成的），前面的是控制信息，数据部分为空
2. 服务器对连接请求回应，同时含有连接请求信息（ack#=client seq# +1，server seq#），设置了标志（syn=1，ack=1），数据部分为空
3. 客户端将ack标志设为1，ack#=server seq#+1，数据中包含有请求的数据
4. 服务器从文件系统中取出相应内容进行打包，返回给客户端（ack=1，server seq#）
5. 关闭连接（四步实现）

1-3进行的操作称为三次握手，实际上是客户端和服务器进行信息交换，信息交换的目的是客户端和服务器端进行数据初始化和分配缓存，由操作系统内部实现，向用户提供API。

第五步就是缓存的释放，也是通过信息交换来实现。由用户程序来控制。

二、HTTP消息格式（关注的是c/s端传输的应用消息，忽略TCP消息）

1、Request请求消息（ASCII编码，定义了控制字符、标点符号、数字、字母、运算符、空格），HTTP编码的没有控制字符。除了控制字符以外的字符都是人们可以识别的字符。

1）、命令行（只有一行），以\r\n结束

GET、POST、HEAD、PUT、DELETE

命令 目标 协议/版本

2）、头行（多行）Header,行与行之间以\r\n来间隔，头行结束以\r\n\r\n标示

User-agent，Host，Authorization等等，header line除了一些browser/server由约定的，还可以进行任意扩展。

Header line的组成：

Key：value

3）、数据体（多行）（Body）

注意GET命令没有数据体

使用telnet调试，telnet可以作为客户段使用

请求格式是将数据和命令一起编码，靠规定格式来分隔数据和命令

命令的用途：

GET用来获取服务器上不变的内容

POST用来在服务器上更新内容，更新的内容放在body，数据通过body传递

PUT在服务器上创建新的对象，创建的对象放在Body，数据通过body传递

DELETE删除服务器上的对象

数据库应用时使用SQL命令（语句），实际使用SQL构造crud，什么是crud？Create，read，update，delete。

CRUD可以分为两类：Command（Create、update、delete）和Query（read）。

一类需要对服务器端进行更改，更新数据的频度低，

一类仅查询，查询不改变服务器端的数据，读数据的频度高，就可以把更新和query分开，仅对主备份进行修改，从备份用于query，如果一个服务器不能承担可以使用多个服务器，主副本和从副本之间采用最终一致性方法实现对从副本的更新。

CQRS—大家自己去查询。--今天的作业：搞清楚CQRS的优缺点，要求按照论文格式写，要求有题目、摘要、内容，字数>2000字。题目自拟。

2、Response响应消息

1）、状态行

协议名称/版本号 状态码 状态短语

编程时只需要判断状态码就可以，状态码和状态短语是相互冗余的

2）、头行（首部行）

3）、Body，传递命令执行结果

状态码必须熟知，microservices，大家在使用Microservices构建分布式应用时还得熟悉更多的状态码。

4、构建支持持久状态—Cookie/Session实现

1）、浏览器支持Cookie

* 接收到Set-cookie头行时，就把Set-cookie的内容存入本地
* 相服务器发出命令时要将相应站点的cookie附上作为头行传递给服务器

2）、服务器端

接受到的命令中无cookie时自动生成cookie id，有的话，就讲相应的请求需要的变量和数据库中相应的变量表示的资源合并保存在数据库中，每次应答时都声称Set-cookie头行返回给客户端。

3）、cookie用途

授权，服务器接受到的请求没有cookie或cookie过期时，转移到登录界面，用户登录，登录时输入用户名和口令传给服务器，服务器进行验证，验证通过后穿件cookie id，将cookie ID存入数据库，并以Set-cookie作为返回。

购物车：将选择的物品放入购物车就是将cookie和所选物品传给服务器，服务器就追加物品或删除。

1. Web缓存

架设Proxy服务器，向web服务器的请求不是直接想web服务器请求，而是向proxy服务器请求，由proxy服务器决定下一步动作，首先在本地缓存中查找有没有副本，有副本的话就用副本应答，没副本向web服务器请求，请求的结果先存为副本，再用副本应答。需要解决：

* 1. 如何确定本地副本，采用key来查找，key就是地址，value内容
  2. 如何解决服务器端更新，条件GE，利用头行解决If-modified-since，头行包括本地副本的修改时间，向web服务器请求，web服务器将该时间和真实副本修改时间进行比较，副本更新的话就传新的副本，否则回复没有更新。Proxy服务器做相应处理。

作用：

User-agent：就是浏览器

当前使用的架构是Microservices，RESTFul web services，分布式架构，当时采用的通信协议是HTTP协议，采用HTTP协议的原因：

简单、不用另构造、适合FireWall来防范。

2.3 电子邮件

SMTP/POP3/IMAP

三个主要功能模块

1. 用户端软件—user agent
   1. 起草、阅读、修改邮件
   2. 从邮件服务器获取邮件
   3. 发送邮件
   4. 充当UA的软件：
      1. 专用的客户端软件，如Outlook等
      2. 通用的软件，浏览器
2. 邮件服务器
   1. 为用户存储邮件
      1. 为每个用户设置邮箱
      2. 为某个用户接收邮件，并将接收到的邮件存入用户邮箱里
   2. 接受邮件（SMTP）
      1. 一个来自于user agent—存入发送队列
      2. 接受来自其它邮件服务器发送过来的邮件
   3. 发送邮件（SMTP）
      1. 从发送队列取邮件，根据邮件接收地址发送给相应服务器
3. SMTP协议（Simple Message Transfer Protocol）

协议端口号：25

传输方式：User Agent—mail server—Mail Server

1. 邮件地址：用户名@域名
   1. 注册服务器上的用户名
   2. 域名：DNS命名
   3. 注册服务器通常是类UNIX系统，在这些系统上注册一个用户，就为用户创建一个进入该系统账号，并且在用户home目录下创建一个专门存放邮件的文件。只容许用户通过邮件访问协议和SMTP访问该服务器，不能进行远程登录

在配置邮件代理时需配置

* 访问邮箱的方式（IMAP协议还是POP3协议）
* 另外采用安全方式还是明码方式
  + 明码方式存在安全性问题
    - 用户名和密码以及邮件都采用的是明码传输
  + 安全方式
    - 所有传输都采用加密传输
* 还有注册邮箱账号和密码
  + 第一个目的注册邮箱服务器需验证合法用户，在1998年前后发送邮件是不进行合法用户验证，因此邮件的来源没法保证，人们利用这一漏洞产生大量垃圾邮件。
  + UA从邮件服务器获取邮件时进行验证

1. SMTP协议
   1. 两种工作模式
      1. 命令模式，缺省模式，首先进入的模式，传输的是命令
      2. 数据传输模式，模式的转换
         1. 命令模式—数据传输模式的转换，发送DATA命令，接收到DATA命令后就转换为数据传输
         2. 数据传输模式—命令模式转换，发送.\r\n
         3. 数据传输的特点：以行为单位进行传输，每行都是以\r\n进行标示
         4. 传输时一次一封邮件
2. User Agent如何从用户邮箱里获取邮件 （POP3/IMAP），统称邮件访问协议
   1. 如何判断服务器和UA中邮件的重复问题
      1. 第一通过邮件大小进行初步处理
      2. 大小相同的下载下来后进行详细处理
   2. 在服务器端海需服务器软件
      1. 执行POP3协议的开源软件：popper

2.4 DNS（Domain Name Service）

对节点进行命名，解决IP无法记忆问题，采用名字标示节点

名称的唯一：局部、范围甚至全局唯一，名字存在重名问题，DNS引入了层次化命名

引入新的问题：名字和节点地址之间转换问题—使用DNS服务解决，名字—IP之间转换，解决网络基础设施不用更改

层次命名带来新的问题：管理、稳定性、扩展性和效率问题

对名字进行管理，名字和节点之间的对应关系必须由人来手工加入，集中管理还是分散管理？

集中：节点规模让人崩溃

分散：每个范围的规模适中，适合人工管理（X）

分级管理，每级仅管理该级别下的域名

域名服务器存在：

1. 根域名服务器
2. 顶级域名服务器
3. 权威域名服务器
4. 本地域名服务器
5. 域名服务器的作用：
   1. 本地域名服务器，专用于解决域名解析，不承担域名管理，设置在每个ISP
   2. 其它域名服务器，除了解析所管辖范围内域名解析外，还承担对相关级别域名的管理，在每个层级的域名上都要设置相应域名服务器
   3. 如何设置相应级别的域名服务器：在上级域名服务机构注册域名以及相应服务器IP地址。通常设置两个域名服务器，解决单点故障，其中一个用于管理，另一个用于备份服务。

域名解析过程：

当要和某一节点进行通信时，必须告诉本地软件远程节点名称，先查本地缓存中是否有名称和节点地址，没有向本地域名服务器查询，本地域名服务器是在对本机的网络配置时配置的：本机IP地址、网络掩码、缺省网关和本地DNS服务器。

配置方式：1、人工 2、自动（DHCP）

最近新增了顶级域名：io（云计算）、dev（新的开发工具类）

顶级域名服务器：在全世界范围设置了13个，不过在中国没有。

澄清几个问题：

1. 美国关闭顶级域名服务器怎么办？使用（解析域名）是否受限？无法隔离使用权
2. 顶级域名控制的是管理权不是解析，管理权受限存在哪些问题？增加顶级域名权利受限
3. IPv9？IPv9就是使用十进制数字对节点命名，受电话号码的启发
   1. 记忆电话号码？记不住，骗人的
   2. 安全？相当于另起炉灶，自外于整个Internet

域名管理：有专门的文件存放DNS记录

注意的问题：

1. 一个名称对应多个IP地址，用于负载均衡
2. 多个名字对应一个IP地址，用于虚拟服务器
3. 多个名字到多个IP地址，结合上面两个。

写文章作业：

格式必须符合要求：

摘要

绪论

正文：

首先问题的来源

问题的原因

如何解决

解决的效果（结论）

参考文献