# 应用层

## 一、应用层协议原理

网络应用是计算机网络存在的理由。

研发网络应用程序的核心是写出能够运行在不同端系统和通过网络彼此通信的程序，当研发新应用程序时，需要编写将在多台端系统上运行的软件，不需要写在网络核心设备如路由器或链路层交换机上运行的软件。即使要为网络核心设备写一个应用程序，也不可能做到这一点。网络核心设备并不在应用层上起作用，特别是在网络层及下面层次起作用。这种基本设计，将应用软件限制在端系统的方法，促进了大量网络应用程序的迅速研发和部署。

### 1. 网络应用程序体系结构

应用程序的体系结构明显不同于网络的体系结构，从应用程序研发者的角度来看，网络体系结构是固定的，并为应用程序提供了特定的服务集合。

\*\*应用程序体系结构\*\*由应用程序的研发者设计，规定了如何在各种端系统上组织该应用程序。现代网络应用程序中所使用的两种主流体系结构之一：\*\*客户-服务器体系结构\*\*或\*\*对等(P2P)体系结构\*\*

\*\*客户-服务器体系\*\*：利用客户-服务器体系结构，客户之间互相之间不直接通信，例如，在Web应用中两个浏览器并不直接通信。客户-服务器体系结构的另一个特征是该服务器具有固定的，众所周知的地址，该地址被称为\*\*IP地址\*\*。由于该服务器地址固定，且总是打开，客户总是可以通过该服务器的IP地址来发送分组与其联系。

在一个客户-服务器应用中，常常会出现一台单独的服务器主机跟不上它所有的客户请求的情况。为此，配备大量主机的数据中心常被用于创建强大的虚拟服务器。服务提供商必须支付不断出现的互联和带宽费用，以发送和接收到达/来自数据中心的数据。

\*\*P2P体系\*\*：在一个P2P体系结构中，对于数据中心的专用服务器有最小的(或者没有)依赖。相反，应用程序在间断连接的主机对之间使用直接通信，这些主机被称为\*\*对等方\*\*，对等方通信不需要铜鼓哦专门的服务器。许多目前流行的，流量密集型应用都是P2P体系结构的(包括迅雷，skype等)。某些应用具有混合的体系结构，它结合了客户-服务器体系结构和P2P体系，如在用于即使讯息应用中，服务器被用于跟踪用户的IP地址，但用户到用户的报文在用户主机之间直接传输。

P2P体系结构最引人入胜的特性是它们的\*\*自扩展性\*\*。它们通常不需要庞大的服务器基础设施和服务器带宽。但是P2P应用由于高度的非集中式结构，面临安全性，性能和可靠的挑战。

### 2. 进程通信

进行通信本质上是\*\*进程\*\*，而不是程序。一个进程可以被认为是运行在端系统中的一个程序，当多个进程运行在相同的端系统上时，它们使用进程间通信机制相互通信。进程间通信的规则由端系统上的操作系统确定。

在两个不同端系统上的进程，通过跨越计算机网络交换\*\*报文\*\*而相互通信，发送进程生成并向网络中发送报文，接收进程则接收这些报文并可能通过回送报文进行响应。

\*\*客户和服务器进程\*\*：网络应用程序由成对的进程组成，这些进程通过网络相互发送报文。在每队通信进程中，我们常将这两个进程之一标识为\*\*客户\*\*，另一个标识为\*\*服务器\*\*。对于Web而言，浏览器是一个客户进程，Web服务器是一台服务器进程。对于P2P文件共享，下载文件的对等方标识为客户，上传文件的对等方表示为服务器。

\*\*进程与计算机网络之间的接口\*\*：进程通过一个称为\*\*套接字\*\*的软件接口向网络发送报文和从网络接收报文。套接字是一台主机内应用层和运输层之间的接口，由于该套接字是建立在网络应用程序的可编程接口，因此套接字称为应用程序和网络之间的\*\*应用程序编程接口(API)\*\*。

应用程序的开发者可以控制套接字在应用层端的一切，但是对于该套接字的运输端几乎没有控制权。

\*\*进程寻址\*\*：在一台主机上运行的进程为了向另一台主机上运行的进程发送分组，接收进程需要有一个地址。为了标识该进程，需要定义两种信息，一是主机的地址，二是在目的主机中指定接收进程的标识符。

在因特网中，主机由其IP地址标识，IP地址是一个32比特的量且它能够唯一的标识该主机。除了直到所发送的报文的目的地外，发送进程还需要指定运行在接收主机上的接收进程(接收套接字)。\*\*目的地端口号\*\*用于这个目的，大部分的应用已经分配了特定的端口号。

### 3.可供应用程序使用的运输服务

套接字是应用程序和运输层协议之间的接口，在发送端的应用程序将报文推进该套接字。在该套接字的另一侧，运输层协议负责从接收进程的套接字得到该报文。包括因特网在内的很多网络提供了不止一种运输层协议，当开发一个应用时，必须选择一种可用的运输层协议。

一个运输层协议大体提供四种服务：\*\*可靠数据服务\*\*，\*\*吞吐量\*\*，\*\*定时\*\*，\*\*安全性\*\*。

\*\*可靠数据服务\*\*：分组在计算机中可能会丢失，如在路由器中的缓存溢出，或者当分组中某些比特损坏后可能会被丢弃。像电子邮件，金融应用，Web文档传输这些，必须要将数据完整，正确的交付给目的地。如果一个协议能做好这件事，就认位它提供了可靠数据服务。当一个运输协议提供这种服务时，发送进程只要将其数据传递进套接字，就可以完全相信该数据将能无差错的到达接收进程。

当一个运输层协议不提供可靠数据服务时，某些数据可能不能到达目的地。这可能能被\*\*容忍丢失的应用\*\*所接收，如交谈式音频，视频，它们可以承受一定量的数据丢失。

\*\*吞吐量\*\*：即运输层协议能够以某种特定的速率提供确保的可用吞吐量。具有吞吐量要求得应用程序被称为\*\*带宽敏感得应用\*\*，许多目前得多媒体应用是带宽敏感的。带宽敏感的应用具有特定的吞吐量要求，而\*\*弹性应用\*\*能够根据当时可用的带宽或多或少的利用可供使用的吞吐量。如电子邮件，文件传输，Web传送都属于弹性应用。当然吞吐量越多越好

\*\*定时\*\*：运输层协议也能提供定时保证，这种服务对交互式实时应用程序有吸引力，如网络电话，虚拟环境，多方游戏。这些服务为了有效性而要求数据交付有严格的时间限制。

\*\*安全性\*\*：运输层协议可以为应用程序提供一种或多种安全服务，如在发送主机中，运输协议能加密由发送进程传输的所有数据，在接收进程中，运输层协议能将数据解密后再交给接收进程。运输协议除了提供机密性的安全服务，还可以鉴定数据完整性和端点鉴别

### 4. 因特网提供的运输服务

因特网(TCP/IP网络)为应用提供了两个运输层协议，即UDP和TCP

\*\*TCP服务\*\*：TCP服务模型包括面向连接服务和可靠数据传输服务，当某个应用程序调用TCP作为运输协议时，该程序可以使用TCP提供的这两种服务。

\*\*面向连接的服务\*\*：在应用层数据报文开始流动之前，TCP让客户和服务器互相交换运输层控制信息，这个所谓的握手过程提醒客户和服务器，让它们为大量分组的到来做好准备。在握手阶段，一个TCP连接就在两个进程的套接字之间建立了，这条连接是全双工的(双方进程可以在此连接上同时进行报文收发)

\*\*可靠数据传输服务\*\*：通信进程能依靠TCP，无差错，按适当顺序交付所有发送的数据。

\*\*TCP拥塞控制机制\*\*：当发送方和接收方之间的网络出现拥塞时，TCP拥塞控制机制会抑制发送进程，TCP拥塞控制机制也会试图限制每个TCP连接，使它们达到公平共享网络带宽的目的

\*\*TCP安全\*\*：无论是TCP还是UDP都没有提供任何的加密机制，因特网界研发出了TCP的加强版\*\*安全套接字层 SSL\*\*，SSL不但能完成TCP可以完成的，还提供了进程到进程的安全性服务，包括加密，数据完整性，端点鉴别。SSL不是一种和TCP，UDP在同一层次上的第三种运输协议，而是一种对TCP的加强，这种强化是在应用层上实现的。

\*\*UDP服务\*\*：UDP是一种不提供不必要服务的轻量级运输协议，它仅提供最小服务。UDP是无连接的，没有握手过程。UDP协议提供的是一种不可靠数据传送服务，UDP也没有拥塞控制机制，所以UDP的发送端可以用它选择的任何速率向其下层(网络层)注入数据。

\*\*因特网运输协议所不提供的服务\*\*：在TCP和UDP的讨论中都没有说到吞吐量和定时时延，即这些服务，目前因特网运输协议并没有提供。总之，今天的因特网通常能为时间敏感的应用提供满意的服务，但它不能提供任何定时或带宽保证。

对于电子邮件，文件传输，Web都使用TCP，最主要的原因是TCP提供的可靠数据传输服务。但skype这样的因特网电话应用的开发者更愿意将该应用运行在UDP上，从而避开TCP的拥塞控制机制和分组开销，这些应用即使丢失了一些数据，也无伤大雅。

### 5. 应用层协议

\*\*应用层协议\*\*：定义了不同端系统上的应用程序进程如何相互传递报文。

有些应用层协议定义在公共区域中，如Web的应用层协议\*\*超文本传输协议 HTTP\*\*。如果浏览器开发者遵从HTTP RFC 规则，那么开发出的浏览器即可访问任何遵循该文档的Web服务器，并获取希望的Web页面。

应用层协议只是网络应用的一部分,Web是一种客户-服务器应用，它允许客户按照需求从Web服务器获得文档。该Web应用由许多部分组成，包括

\*\*文档格式标准 HTML\*\*，Web浏览器，Web服务器，以及一个应用层协议

Web的应用层协议是HTTP，它定义了在浏览器和Web服务器之间传输的报文格式和序列。因此HTTP只是Web应用中的一部分。

## 二、Web 和 HTTP

对于大部分用户来说，Web的按需操作很吸引人。超链接和搜索引擎帮助我们在Web站点的海洋里导航

### 1. HTTP概况

\*\*HTTP\*\*：Web的应用层协议是\*\*超文本传输协议\*\*，它是Web的核心。HTTP由两个程序实现，一个客户程序和一个服务器程序。客户程序和服务器程序运行在不同的端系统中，通过交换HTTP报文进行会话。HTTP定义了这些报文的结构以及客户和服务器进行报文交换的方式。

\*\*Web页面\*\*：也叫文档，是由对象组成的，一个对象只是一个文件，如一个HTTP文件，一个JPEG图形，等这样的一个文件。且它们都可以通过一个\*\*URL\*\*地址寻址。多数Web页面含有一个HTML基本文件以及几个引用对象。如一个Web页面包含了HTML文本和5个JPEG图形，那么这个Web页面有6个对象。HTML基本文件通过对象的URL地址引用页面中的其他对象。

每个URL由两部分组成，\*\*存放对象的服务器名 和 对象的路径名\*\*。

Web浏览器实现了HTTP的客户端，Web服务器实现了HTTP的服务器端，它用于存储Web对象，每个对象用URL寻址。

HTTP定义了Web客户向Web服务器请求Web页面的方式，以及服务器向客户传送Web页面的方式

HTTP使用TCP作为它的支撑运输协议，TCP为HTTP提供可靠的数据传输服务。HTTP协议不用担心数据丢失，也不关注TCP以及协议栈较低层的工作

客户端的套接字接口是客户进程与TCP连接之间的门，在服务器端的套接字接口则是服务器进程与TCP连接之间的门。一旦客户向他的套接字发送一个完整的报文，该报文就脱离了客户控制并进入TCP控制。

### 2. 非持续连接和持续连接

在许多因特网应用程序中，客户和服务器在一个相当长的时间内通信，其中客户发出一系列请求并且服务器对每个请求进行响应。当这种客户-服务器的交互是经TCP进行时，要确定每个请求/响应是由一个单独的TCP连接发送还是所有的请求及其响应由相同的TCP连接发送。前者被称为\*\*非持续连接\*\*，后者被称为\*\*持续连接\*\*。HTTP既能使用非持续连接，也能够使用持续连接，尽管HTTP在默认方式下使用持续连接，HTTP客户和服务器也能配置成使用非持续连接。

\*\*采用非持续连接的HTTP\*\*：每个TCP连接在服务器发送一个对象后关闭，即该连接并不为其他的对象而持续下来。每个TCP连接只传输一个请求报文和一个响应报文。\*\*往返时间 RTT\*\*，指一个短分组从客户到服务器然后再返回客户所花费的时间。RTT包括分组传播时延，排队时延及分组处理时延。当一个用户点击一个超链接时，浏览器在它和Web服务器之间发起一个TCP连接，这涉及到依次“三次握手”的过程，即客户向服务器发送一个小TCP报文段，服务器用一个小TCP报文段做出确定和响应，最后，客户向服务器返回确定。粗略来讲，总的响应时间是两个RTT加上服务器传输HTML文件的时间。

\*\*非持续连接缺点\*\*：第一，必须为每一个请求的对象建立和维护一个全新的连接，对于每个这样的连接，在客户和服务器中都要分配TCP的缓冲区和保持TCP变量，这给Web服务器带来了严重的负担。第二，每一个对象经受两倍RTT的交付时延。

\*\*采用持续连接的HTTP\*\*：在使用HTTP1.1持续连接的情况下，服务器在发送响应后保持该TCP连接打开，在相同的客户机与服务器之间，后续的请求和响应报文能通过相同的连接进行传送。

### 3. HTTP报文格式

HTTP规范包含了对HTTP报文格式的定义，HTTP报文有两种，请求报文和响应报文。

\*\*HTTP请求报文\*\*：

![在这里插入图片描述](https://img-blog.csdnimg.cn/20200813110512445.png?x-oss-process=image/watermark,type\_ZmFuZ3poZW5naGVpdGk,shadow\_10,text\_aHR0cHM6Ly9ibG9nLmNzZG4ubmV0L3dlaXhpbl80MzU0MTA5NA==,size\_16,color\_FFFFFF,t\_70#pic\_center)

\*\*HTTP响应报文\*\*：

![在这里插入图片描述](https://img-blog.csdnimg.cn/20200813111836889.png?x-oss-process=image/watermark,type\_ZmFuZ3poZW5naGVpdGk,shadow\_10,text\_aHR0cHM6Ly9ibG9nLmNzZG4ubmV0L3dlaXhpbl80MzU0MTA5NA==,size\_16,color\_FFFFFF,t\_70#pic\_center)

### 4. 用户与服务器的交互 cookie

HTTP服务器是无状态的，然而Web站点确希望能够识别用户，可能因为服务器限制用户的访问，或它希望把内容和用户身份联系起来，为此HTTP使用了cookie，它允许站点对用户跟踪，大多数商务站点都使用了cookie

\*\*cookie\*\*：cookie技术由四个组件构成：1，在HTTP响应报文中的一个cookie首部行，2，在HTTP请求报文中的一个cookie首部行，3，在用户端系统中保留一个cookie文件，由用户浏览器管理，4，位于Web站点的一个后端数据库。

cookie可以用于标识一个用户，用户首次访问时，可能需要提供一个用户标识，在后继会话中，浏览器向服务器传递一个cookie首部，从而向该服务器标识了用户。因此cookie可以在无状态的HTTP上建立一个用户会话层。

### 5. Web缓存

Web缓存器也叫\*\*代理服务器\*\*，它是能够代表初始Web服务器来满足HTTP请求的网络实体。Web缓存器有自己的磁盘管理空间，并在存储空间中保存最近请求过的对象的副本。

![在这里插入图片描述](https://img-blog.csdnimg.cn/20200813113557355.png?x-oss-process=image/watermark,type\_ZmFuZ3poZW5naGVpdGk,shadow\_10,text\_aHR0cHM6Ly9ibG9nLmNzZG4ubmV0L3dlaXhpbl80MzU0MTA5NA==,size\_16,color\_FFFFFF,t\_70#pic\_center)

\*\*Web缓存\*\*：Web缓存器既是服务器也是客户，当他接收浏览器的请求并发回响应时，它是服务器，当它向初始服务器发出请求并接收响应时，它是一个客户。部署Web缓存器的两个原因，首先，Web缓存器可以大大减少对客户请求的响应时间。其次，Web缓存器能从整体上大大减低因特网上的Web流量，改善应用的性能。

通过使用\*\*内容分发网络 CDN\*\*，Web缓存器正在因特网中发挥越来越重要的作用。

### 6. 条件GET方法

尽管高速缓存能减少用户感受到的响应时间，但也引入了一个新的问题，即存放在缓存器中的对象副本可能是陈旧的。HTTP协议有一种机制，允许缓存器证实它的对象是最新的，这种机制就是\*\*条件GET\*\*。

\*\*条件GET\*\*：将一个代理缓存器代表一个请求浏览器，向某Web发送一个请求报文，方法字段采用GET来检查某对象是否被更改过了。GET报文高速服务器，仅当自指定日期后该对象被修改过，才发送该对象。如果该对象没有修改，那么Web服务器向缓存器发送一个响应报文，但不会在该响应报文中包含所请求的对象。包含请求对象只会浪费带宽，并增加用户感受到的响应时间。响应报文中的状态行304告诉缓存器可以继续使用该对象，能向请求的浏览器转发该对象在代理缓存器中的副本。

## 三、因特网中的电子邮件

现代电子邮件有许多强大的特性，包括附件，超链接，HTML格式文本和图片的报文。因特网电子邮件系统由三个主要部分组成，\*\*用户代理\*\*，

\*\*邮件服务器\*\*，\*\*简单邮件传输协议 SMTP\*\*。

\*\*用户代理\*\*：允许用户阅读，回复，转发，保存和撰写报文。

\*\*邮件服务器\*\*：形成了电子邮件体系的核心，每个接收方在其中某个邮件服务器上有一个\*\*邮箱\*\*。一个典型的邮件发送过程：从发送方的用户代理开始，传输到发送方的邮件服务器，再传输到接收方的邮件服务器，然后在这里被分发到接收方的邮箱里。

\*\*SMTP\*\*：是因特网电子邮件中主要的应用层协议。它使用TCP可靠数据传输服务，SMTP也有两个部分，运行在发送方邮件服务器的客户端和运行在接收方邮件服务器的服务器端。每台邮件服务器上既运行SMTP的客户端，也运行SMTP的服务器端。

### 1. SMTP

SMTP是用于从发送方的邮件服务器发送报文到接收方的邮件服务器。SMTP一般不使用中间邮件服务器发送邮件，即使这两个邮件服务器位于地球的两端也是这样。

\*\*发送过程\*\*：用户在用户代理上写好报文后，用户代理将报文发给用户的邮件服务器，在那里报文被放在报文队列中。运行在发送方上的邮件服务器上的SMTP客户端发现这个报文队列中的报文，它就创建一个到接收方邮件服务器上的SMTP服务器的TCP连接。在接收方的邮件服务器上，SMTP的服务器端接收该报文，接收方的邮件服务器将该报文放到接收方的邮箱中。在接收方方便的时候，他可以调用用户代理阅读该报文。

\*\*SMTP工作原理流程\*\*

首先，客户SMTP在25号端口建立一个到服务器SMTP的TCP连接。如果服务器没有开机，客户会在稍后继续尝试连接。一旦建立连接，服务器和客户执行某些应用层的握手，就像人与人间交流前的自我介绍。SMTP的客户和服务器在传输信息前先相互介绍。在SMTP握手阶段，SMTP客户指示发送方的邮件地址和接收方的邮件地址。一旦该SMTP客户和服务器彼此介绍后客户就发送该报文。SMTP能依赖TCP提供可靠的数据传输无差错的将邮件投递到接收服务器。该客户如果有另外的报文要发送到该服务器，就在该相同的TCP连接上重复这种处理；否则，他指示TCP关闭连接。

### 2. 与HTTP的对比

HTTP和SMTP这俩个协议都用于从一台主机向另一台主机发送文件，HTTP从Web服务器向Web客户(通常为浏览器)传送文件(对象)。SMTP从一个邮件服务器向另一个邮件服务器传送文件(电子邮件报文)。

当进行文件传输时，持续的HTTP和SMTP都使用持续连接，因此这两个协议有一些共同特征，但也有一些区别。

首先，HTTP主要是一个\*\*拉协议\*\*，TCP连接是由想接收文件的机器发起的。SMTP基本上是一个\*\*推协议\*\*，这个TCP连接是由要发送该文件的机器发起的

第二个区别，SMTP要求每个报文采用7比特ASCII码格式，HTTP数据不受这种限制。

第三个区别是如何处理一个既包含文本又包含图形的文档，HTTP将每个对象封装到它的HTTP响应报文中，而SMTP将所有的报文对象放在一个报文中

### 3. 邮件报文格式（与SMTP命令区分开）

\*\*邮件报文格式\*\*：当一个人给另一个人发送电子邮件时，一个包含环境信息的首部位于报文体前面。这些报文信息包括在一系列首部中，如同HTTP协议，每个首部行包含了可读的文本，每个首部行必须有一个From ：首部行和一个To：首部行。在报文首部行后，紧接着一个空白行，然后是以ASCII码格式表示的报文体。

### 4. 邮件访问协议

![在这里插入图片描述](https://img-blog.csdnimg.cn/20200813152430916.png#pic\_center)

接收方的用户代理不能使用SMTP得到报文，因为取报文是一个\*\*拉操作\*\*

而SMTP是一个\*\*推协议\*\*，为了解决这个问题，通过引入一个特殊的邮件访问协议来解决。目前流行的邮件访问协议包括，\*\*第三版的邮局协议POP3\*\*

\*\*因特网邮件访问协议 IMAP\*\*，以及HTTP

\*\*POP3\*\*：POP3是一个极为简单的邮件访问协议，当用户代理打开了一个到邮件服务器的TCP连接后，POP3开始工作，POP3按照三个阶段工作\*\*特许\*\*，\*\*事务处理\*\*，\*\*更新\*\*。

\*\*POP3工作流程及原理\*\*：在第一个阶段，即特许阶段，用户代理发送用户名和口令鉴别用户。在第二阶段，即事务处理阶段，用户代理取回报文；同时在这个阶段用户代理还能进行如下操作，对报文作删除标记，取消报文删除标记，以及获取邮件的统计信息。在第三个阶段即更新阶段，他出现在客户发出quit命令以后，目的是结束该POP3会话；这时，该邮件服务器删除那些被标记为删除的报文。

\*\*IMAP\*\*：由于POP3协议没有给用户提供任何创建远程文件夹并为报文指派文件夹的方法。IMAP应运而生，IMAP服务器把每一个报文与一个文件夹关联起来，可以从任何一台机器对所有的报文进行访问。

\*\*IMAP功能\*\*：IMAP协议为用户提供了创建文件夹以及将邮件从一个文件夹移动到另一个文件夹的命令。IMAP还为用户提供了在远程文件夹中查询邮件的命令，按指定条件去查询文件。IMAP的另一个重要特性是它具有允许用户代理获取报文某些部分的命令。

## 四、因特网的目录服务

因特网上的主机和人类一样，可以用多种方式进行标识。主机的一种标识方法是用它的\*\*主机名(如 www.facebook.com)\*\*，然而主机名几乎没有提供关于主机在因特网中的位置信息。可能某些主机名以 .cn 结尾，这仅仅告诉我们该主机在中国而已，况且，主机名可能由不定长的数字字母组成，路由器难以处理。由于这些原因，主机也可以使用所谓的IP地址进行标识。

\*\*IP地址\*\*：一个IP地址由4个字节组成，并有着严格的层次结构，

如121.7.106.83这样一个IP地址，其中灭个字节都被一个 . 分开，表示了

0~255的十进制数字，我们说IP地址具有层次结构，是因为我们从左到右扫描它时，我们会得到越来越具体的关于主机位于因特网何处的信息(即在众多网络的哪个网络里)

### 1. DNS提供的服务

识别主机有两种方式，通过主机名或者IP地址，人们喜欢便于记忆的主机名标识方式，而路由器喜欢定长的，有层次结构的IP地址。为了折中这些偏好，我们需要一种能进行主机名到IP地址转换的目录服务，这就是\*\*域名系统 DNS\*\*的主要任务

\*\*DNS\*\*：DNS是，1，一个由分层的DNS服务器实现的分布式数据库，

2，一个使得主机能够查询分布式数据库的应用层协议，DNS协议运行在UDP上，使用53号端口

DNS是通过客户-服务器模式提供的重要网络功能，DNS不是一个直接和用户打交道的应用，相反DNS是为因特网上的用户应用程序以及其他软件提供一种核心功能，即\*\*将主机名转换成背后的IP地址\*\*。DNS通常是由其他应用层协议所使用的，包括HTTP，SMTP，FTP，将用户提供的主机名解析为IP地址。

\*\*通过DNS获得某主机的IP地址的过程\*\*：1，用户主机上运行着DNS应用的客户端，2，浏览器从URL中抽取目的地主机名，并将其传给DNS应用的客户端，3，DNS客户向DNS服务器发送一个包含主机名的请求，4，DNS客户向DNS服务器发送一个包含主机名的请求，5，一旦浏览器接收到来自DNS的该IP地址，它能够向位于该IP地址80端口的HTTP服务器进程发起一个TCP连接。

除了进行主机名到IP地址的转换外，DNS还提供了一些重要的服务：

\*\*主机别名\*\*：有着复杂主机名的主机可能拥有多个别名，主机别名比主机名更容易记忆，应用程序可以调用DNS来获得主机别名对应的规范主机名以及主机的IP地址。

\*\*邮件服务器别名\*\*：电子邮件应用程序也可以调用DNS，对提供的主机名别名进行解析，以获得该主机的规范主机名及其IP地址。

\*\*负载分配\*\*：DNS也用于在冗杂的服务器之间进行负载分配。

### 2. DNS工作机理概述

DNS是一个提供简单，直接的转换服务的黑盒子，事实上，实现这个服务的黑盒子十分复杂，它由分布于全球的大量DNS服务器以及定义了DNS服务器与查询主机通信方式的应用层协议组成。

DNS的一种简单设计是在因特网上只使用一个DNS服务器，该服务器包含所有的映射，在这种集中式设计中，所有客户直接将查询发往单一的DNS服务器，且该DNS服务器要对每个请求做出响应，虽然这种模式很简单，但它并不适用于今天的因特网，因为因特网有着数量巨大(且持续增长的主机)，这种集中式设计也会出现很对问题：

\*\*单点故障\*\*：如果该DNS服务器故障，整个因特网随之瘫痪

\*\*通信容量\*\*：单个DNS服务器不得不处理所有的DNS查询

\*\*远距离的集中式数据库\*\*：单个DNS服务器无法邻近所有的查询客户，如果将单台DNS服务器放置到巴黎，一个中国的客户想要获得DNS服务查询某主机的IP地址，那么发送和接收的过程要经过很长的链路，也许还伴随着链路的低速和拥塞，这将导致严重的时延

\*\*维护\*\*：单个DNS服务器要为所有的因特网主机保留记录，这不仅使得这个中央数据库庞大，而且还得为每个新出现的主机频繁更新

\*\*根DNS服务器\*\*：有400多个根服务器遍布全世界，由不同的组织管理，根DNS服务器提供TLD服务器的IP地址

\*\*顶级域(TLD)DNS服务器\*\*：对于每个顶级域(org,edu,com,net)和所有的国家级顶级域(uk,fr,cn,jp)都有TLD服务器，TLD服务器提供了权威DNS服务器的IP地址

\*\*权威DNS服务器\*\*：在因特网上具有公共可访问主机(Web服务器和邮件服务器)的每个组织必须提供公共可访问的DNS记录，这些记录将这些主机的名字映射为IP地址

\*\*分布式，层次数据库\*\*：为了处理扩展性问题，DNS使用了大量的DNS服务器，它们以层次方式组织并分布在全世界范围内。大致有三种DNS服务器，\*\*根DNS服务器\*\*，\*\*顶级域(TLD)DNS服务器\*\*，\*\*权威DNS服务器\*\*

客户需要xxx.com的IP地址，首先与根服务器之一联系，它将返回顶级域名(com,org,edu)的TLD服务器的IP地址，该客户则与这些TLD服务器之一连接，它将为xxx.com返回权威服务器的IP地址，最后该客户与xxx.com权威服务器之一联系，它为主机名xxx.com返回其IP地址![在这里插入图片描述](https://img-blog.csdnimg.cn/20200813173653175.png?x-oss-process=image/watermark,type\_ZmFuZ3poZW5naGVpdGk,shadow\_10,text\_aHR0cHM6Ly9ibG9nLmNzZG4ubmV0L3dlaXhpbl80MzU0MTA5NA==,size\_16,color\_FFFFFF,t\_70#pic\_center)

根，TLD和权威服务器都处在该DNS服务器的层次结构中，还有一类很重要的DNS服务器，称为\*\*本地DNS服务器\*\*，一个本地服务器并不属于该服务器的层次结构，但它对DNS层次结构是至关重要的，每个ISP都有一台本地DNS服务器，用户通常能容易的确定本地DNS服务器的IP地址，当本地某主机发出DNS请求时，该请求被发往本地DNS服务器，它起着代理的作用，并将该请求转发到DNS服务器层次结构中

![在这里插入图片描述](https://img-blog.csdnimg.cn/20200813173716634.png?x-oss-process=image/watermark,type\_ZmFuZ3poZW5naGVpdGk,shadow\_10,text\_aHR0cHM6Ly9ibG9nLmNzZG4ubmV0L3dlaXhpbl80MzU0MTA5NA==,size\_16,color\_FFFFFF,t\_70#pic\_center)

上图例子，共发送了8份DNS报文，4份查询报文和4份回答报文，以自己名义来获得映射的叫\*\*递归查询\*\*，而后继的所有查询叫\*\*迭代查询\*\*，因为所有的回答都是直接返回给本地DNS服务器。实践中，查询通常遵守：从请求主机到本地DNS服务器的查询是递归的，其余查询是迭代的。

#### \*\*迭代查询与递归查询\*\*

一、主机向本地域名服务器的查询一般都是采用递归查询，如图b。

​ 所谓递归查询就是：如果主机所询问的本地域名服务器不知道被查询的域名的IP地址，那么本地域名服务器就以DNS客户的身份，

​ 向其它根域名服务器继续发出查询请求报文(即替主机继续查询)，而不是让主机自己进行下一步查询。

​ 因此，递归查询返回的查询结果或者是所要查询的IP地址，或者是报错，表示无法查询到所需的IP地址。

二、本地域名服务器向根域名服务器的查询的迭代查询，如图a。

​ 迭代查询的特点：当根域名服务器收到本地域名服务器发出的迭代查询请求报文时，要么给出所要查询的IP地址，要么告诉本地服务器：“你下一步应当向哪一个域名服务器进行查询”。

​ 然后让本地服务器进行后续的查询。根域名服务器通常是把自己知道的顶级域名服务器的IP地址告诉本地域名服务器，让本地域名服务器再向顶级域名服务器查询。

​ 顶级域名服务器在收到本地域名服务器的查询请求后，要么给出所要查询的IP地址，要么告诉本地服务器下一步应当向哪一个权限域名服务器进行查询。

​ 最后，知道了所要解析的IP地址或报错，然后把这个结果返回给发起查询的主机

![img](%E5%BA%94%E7%94%A8%E5%B1%82.assets/858807-20170820124317600-1783749391.png)

\*\*DNS缓存\*\*：为了改善时延性能并减少因特网上到处传输的DNS报文数量，DNS广泛使用了缓存技术，即在一个请求链中，当某DNS服务器接收一个DNS回答时，它能将映射缓存在本地存储器中。即使它不是该主机名的权威服务器，由于主机和主机名与IP地址间的映射并不是永久的，DNS服务器在一段时间后(通常为两天)将丢弃缓存的信息。本地DNS服务器也能缓存TLD服务器的IP地址，因而允许本地DNS绕过查询链中的根DNS服务器。事实上，因为DNS缓存，除了少数的DNS查询外，根服务器都被绕过了

### 3. DNS记录和报文

共同实现DNS分布式数据库的所有DNS服务器存储了\*\*资源记录 RR\*\*，RR提供了主机名到IP地址的映射，每个DNS回答报文包含了一条或多条\*\*资源记录\*\*，资源记录是一个包含了(Name, Value, Type, TTL)

字段的四元组。\*\*TTL\*\*是该记录的生存时间，它决定了资源记录应当从缓存中删除的时间，\*\*Name和Value\*\*的值取决于\*\*Type\*\*

![在这里插入图片描述](https://img-blog.csdnimg.cn/20200813181511791.png?x-oss-process=image/watermark,type\_ZmFuZ3poZW5naGVpdGk,shadow\_10,text\_aHR0cHM6Ly9ibG9nLmNzZG4ubmV0L3dlaXhpbl80MzU0MTA5NA==,size\_16,color\_FFFFFF,t\_70#pic\_center)

\*\*DNS报文\*\*：DNS有回答和查询报文两种报文，且格式相同

![在这里插入图片描述](https://img-blog.csdnimg.cn/20200813181721104.png?x-oss-process=image/watermark,type\_ZmFuZ3poZW5naGVpdGk,shadow\_10,text\_aHR0cHM6Ly9ibG9nLmNzZG4ubmV0L3dlaXhpbl80MzU0MTA5NA==,size\_16,color\_FFFFFF,t\_70#pic\_center)

\*\*在DNS数据库中插入记录\*\*：如果想注册登记一个新域名，需要在注册登记机构注册新域名，\*\*注册登记机构\*\*是一个商业实体，它验证该域名的唯一性，将该域名输入DNS数据库。因特网名字和地址分配机构向各种注册登记机构授权。当你想注册登记一个新域名时，需要向该机构提供你的基本和辅助权威DNS服务器的名字和IP地址，对这两个权威DNS服务器的每一个，该注册登记机构确保一个类型NS和一个类型A的记录输入TLD服务器

\*\*DNS的安全性\*\*：DNS是因特网基础设施的一个至关重要的组件，对于包括Web和电子邮件等许多重要的服务，没有它都不能正常工作。但DNS对于DDos和带宽泛洪等攻击显示了强大的健壮性，至今还没有一个攻击已经成功妨碍了DNS服务

## 五、P2P文件分发

Web应用，电子邮件和DNS都采用了客户-服务器体系结构，极大的依赖于总是打开的基础设施服务器，使用P2P体系结构，对总是打开的基础设施服务器有最小的(或没有)依赖。P2P体系中成对间歇连接的主机(对等方)彼此直接通信，这些对等方并不为服务提供商所拥有，而是受用户控制的PC

在客户-服务器文件分发中，该服务器必须向每个对等方发送该文件的一个副本，则服务器承受了极大的负担，并且消耗了大量的服务器带宽。而在P2P文件分发中，每个对等方能够向任何其他对等方重新分发它已经收到的该文件的任何部分，从而在分发过程中协助服务器

## P2P体系结构的扩展性

\*\*P2P体系结构的扩展性\*\*：服务器和对等方使用接入链路和因特网相连，对P2P结构进行简单的分析，其中每个对等方能够帮助服务器分发该文件。特别是，当一个对等方接收到某些文件数据，它能够使用自己的上载能力重新将数据分发给其他对等方。具有P2P体系结构的应用程序能够是自扩展的，这种扩展的直接成因是：对等方除了是比特的消费者外还是它们的重新分发者

## BitTorrent

\*\*BitTorrent\*\*：BitTorrent是一种用于文件分发的流行P2P协议，用BitTorrent的话来讲，参与一个特定文件分发的所有对等方的集合被称为一个\*\*洪流\*\*。在一个洪流中的对等方彼此等待下载等长度的文件\*\*块\*\*，典型块长度为256KB，当一个对等方首次加入一个洪流时，它没有块，随着时间的流逝，它积累了越来越多的块，当它下载块时，也为其他对等方上载了多个块。一旦某对等方获得了整个文件，他也许离开洪流，或留在洪流中并继续

向其他对等方上载块

每个洪流具有一个重要的基础设施节点，称为\*\*追踪器\*\*，当一个对等方加入某洪流时，它向追踪器注册自己，并周期性的告诉追踪器它仍在洪流中，以这种方式，追踪器跟踪参与在洪流中的对等方，一个给定的洪流可能在任何时刻具有数以百计千计的对等方

\*\*分布式散列表 DHT\*\*：分布式散列表是一种简单的数据库，其数据库记录分布在一个P2P系统的多个对等方上

## 六、视频流和内容分发网

### 1. 因特网视频

在流式存储视频应用中，基础的媒体是预先录制的视频，这些预先录制好的视频放置在服务器上，用户按需向这些服务器发送请求来观看视频

视频是一系列的图像，通常以一种恒定的速率(如每秒24或30张图像)来展现。一副压缩，数字编码的图像由像素阵列组成，其中每个像素是由一些比特编码来表示亮度和颜色。视频的一个重要特征是它能被压缩，因而可以用比特率来权衡视频质量。比特率越高，图像质量越好，用户的总体视觉感受更好。

到目前为止，对流式视频的最为重要的性能度量是平均端到端吞吐量。为了提供连续不断的布局，网络必须为流式应用提供平均吞吐量，这个流式应用至少与压缩视频的比特率一样大

我们也能使用压缩生成相同视频的多个版本，每个版本有不同的质量等级。

如使用压缩生成同一视频的三个版本，比特率分别为300kbps，1Mbps，3Mbps，用户可以根据他们当前可用带宽来决定看哪个版本

### 2. HTTP流和DASH

在HTTP流中，视频只是存储在HTTP服务器中作为一个普通的文件，每个文件有一个特定的URL，当用户要看此视频时，客户与服务器创建一个TCP连接并发送对该URL的HTTP GET请求。服务器则以底层网络协议和流量条件允许的尽可能快的速率，在一个HTTP响应报文中发送该视频文件。在客户一侧，字节被收集在客户应用缓存中。一旦该缓存中字节数量超过预先设定的门槛，客户应用程序开始播放

虽然HTTP流在实践中已经得到了广泛部署，，但它存在严重缺陷，即所有客户接收到编码相同的视频，这导致了一种基于HTTP的流的研发，它常常被称为\*\*经HTTP的动态适应性流 DASH\*\*，在DASH中，视频编码为几个不同的版本，其中每个版本具有不同的比特率，对应不同的质量水平。客户动态的请求来自不同版本且长度为几秒的视频段数据块，当可用带宽较高时，客户自然的选择高速率版本的块，带宽较低时，就选择低速率版本的块。客户用HTTP GET请求报文一次选择一个不同的块

DASH允许客户使用不同的以太网接入速率流式播放器具有相同编码速率的视频，使用DASH后，每个视频版本存储在HTTP的服务器中，每个版本都有一个不同的URL。HTTP服务器也有一个告示文件，为每个版本提供一个URL及其比特率。DASH允许客户自由地在不同的质量等级间切换

### 3. 内容分发网

YouTube的视频库藏有几亿个，每天向全世界的用户分发几亿条流，向位于全世界的所有用户流式传输所有流量同时提供连续播放和高交互性显然是一项有挑战性的任务

对一个因特网视频公司，提供流式视频服务最为直接的方法是建立单一的大规模数据中心。但该方案存在三个问题，1，如果客户远离数据中心，服务器到客户的分组将跨越许多通信链路并很可能通过很多ISP，如果这些链路之一提供的吞吐量小于视频消耗速率，端到端吞吐量也将小于消耗速率，带来停滞时延。2，流行的视频可能经过相同的通信链路发送许多次，这不仅浪费了网络带宽，因特网视频公司还要为ISP付更多的费用。3，单个数据中心代表一个单点故障，如果数据中心或通向因特网的链路崩溃，它就无法分发任何视频流了

为了应对向全世界的用户分发巨量视频数据的挑战，几乎所有的主要视频公司都利用\*\*内容分发网 CDN\*\*。CDN管理分布在多个地理位置上的服务器，在它的服务器上存储视频(和其他类型的Web内容，包括文档，图片和音频)

CDN可以是\*\*专用CDN\*\*，即它由内容提供商自己所拥有(如谷歌的CDN分发YouTube视频和其他类型的内容)，另一种CDN可以是\*\*第三方CDN\*\*，它代表多个内容提供商分发内容

CDN常采用两种不同的服务器原则：

\*\*深入\*\*：该原则是通过在遍及全球的接入ISP中部署服务器集群来深入到ISP的接入网中，其目标是靠近端用户，通过减少端用户和CDN集群之间链路和路由器的数量，从而改善了用户的感受和时延和吞吐量。因为这种高度分布式设计，使得维护和管理成为挑战

\*\*邀请做客\*\*：通过在少量关键位置建造大型集群来邀请ISP做客，不是将集群放在接入ISP中，这些CDN通常将它们的集群放置在\*\*因特网交换点 IXP\*\*

与深入设计相比，邀请做客产生较低的维护成本和开销，可能以端用户的较高时延和较低的吞吐量作为代价

\*\*CDN操作\*\*：当用户主机中的一个浏览器指令检索一个特定的视频(由URL标识)时，CDN必须截获该请求，以便能够：1，确定此时适用于该用户的CDN服务器集群，2，将客户的请求重定向到该集群的某台服务器

大多数CDN利用DNS来截获和重新定向

\*\*集群选择策略\*\*：任何CDN部署，其核心是集群选择策略，即动态的将客户定向到CDN中的某个服务器或数据中心的机制。为了基于当前流量条件为客户决定最好的集群，CDN能够对其集群和客户之间的时延和丢包性能执行周期性的\*\*实时测量\*\*