访问web页面过程:

1）DNS解析过程

2）HTTP请求响应过程

编写网络嗅探器的流程

1）打开设备（libpcap open）

2）设置过滤器

3）For循环：

a）抓取报文

b）报文解析

1）打开设备（libpcap open, libnet open）

2）设置过滤器

3）For循环：

a）抓取报文

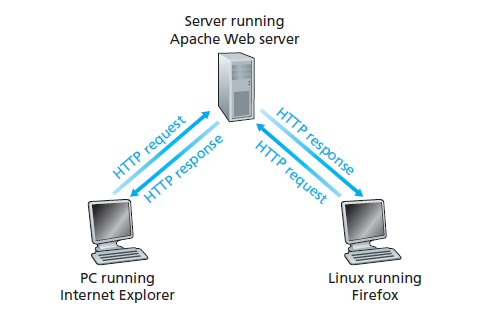
b）报文解析

c）内容匹配

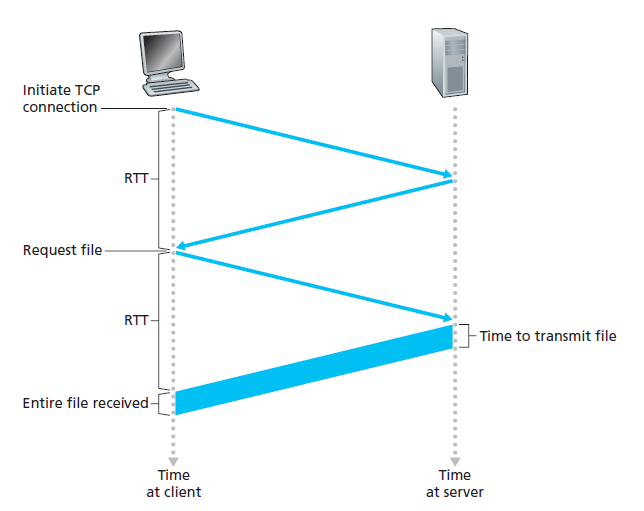
d）伪造响应报文

e）发送响应报文

HTTP的请求-响应行为：



请求并接收一个HTML文件所需的时间估算



典型的HTTP请求报文：

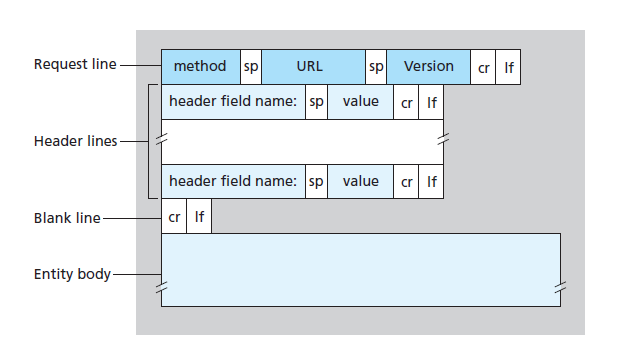
GET /somedir/page.html HTTP/1.1

Host: www.someschool.edu

Connection: close

User-agent: Mozilla/5.0

Accept-language: fr



HTTP/1.1 200 OK

Connection: close

Date: Tue, 09 Aug 2011 15:44:04 GMT

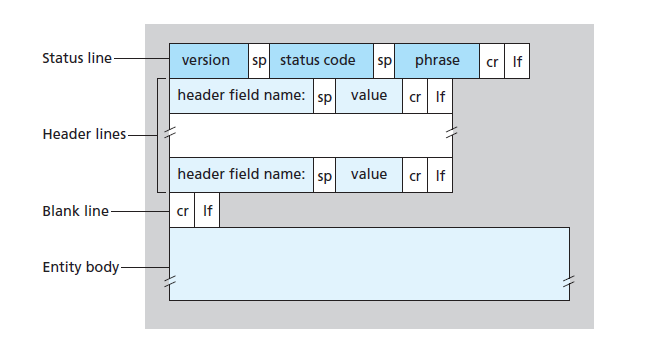
Server: Apache/2.2.3 (CentOS)

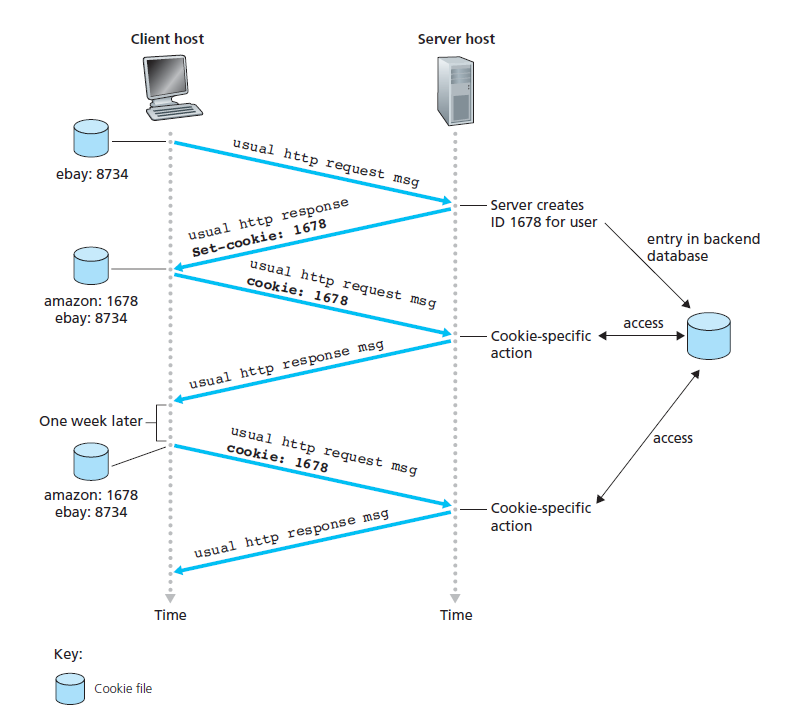
Last-Modified: Tue, 09 Aug 2011 15:11:03 GMT

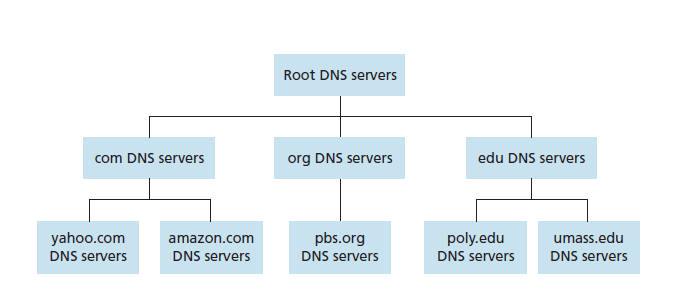
Content-Length: 6821

Content-Type: text/html

(datadatadatadatadata ...)

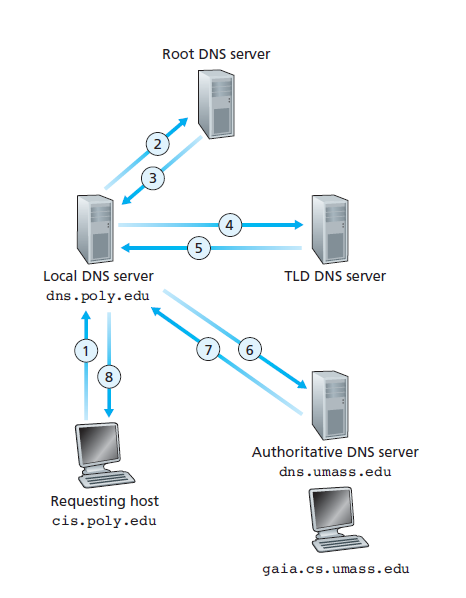






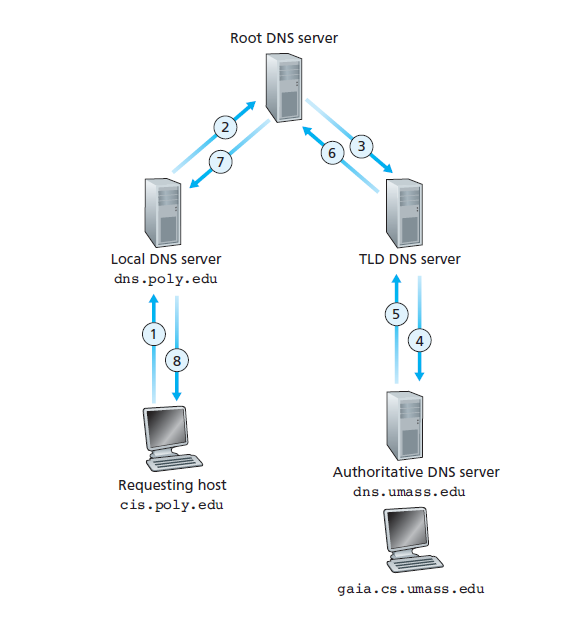
DNS服务实现方式：分布式、层次数据库。

DNS分类：根DNS服务器，顶级域DNS服务器，权威DNS服务器。本地DNS服务器，每个ISP都有一台本地DNS服务器。

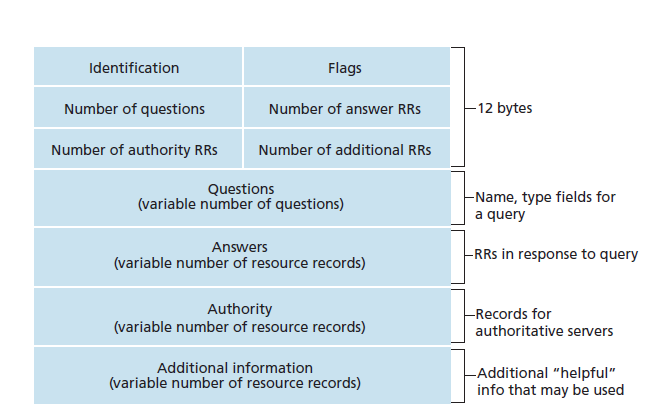


一个简单的域名查找过程：cis.poly.edu访问gaia.cs.umass.edu。

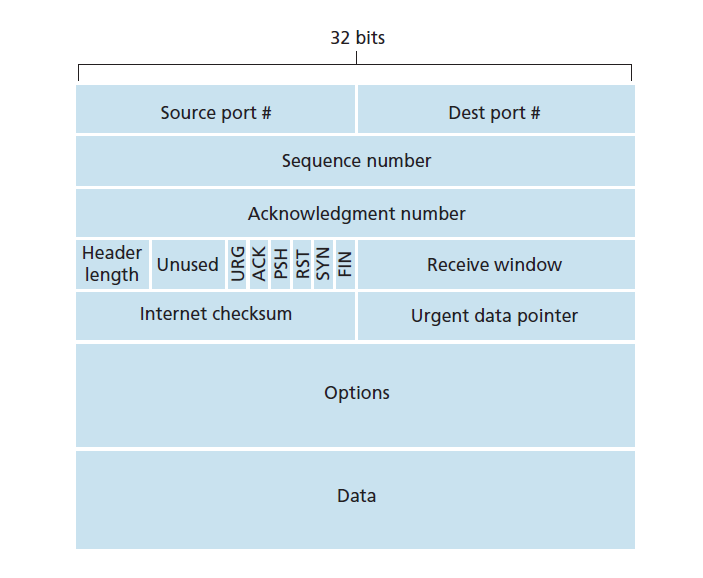
DNS查询方式：递归查询和迭代查询（cis.poly.edu到dns.poly.edu是递归查询），后继3个查询是迭代查询。任何DNS查询既可以是迭代的也可以是递归的。

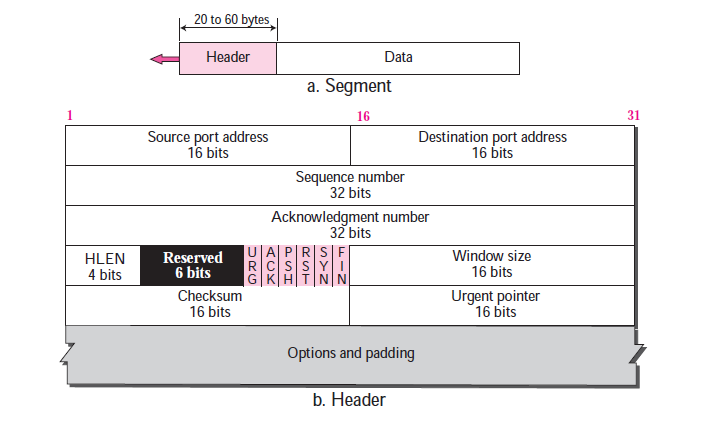


一个完全的递归式的查询。

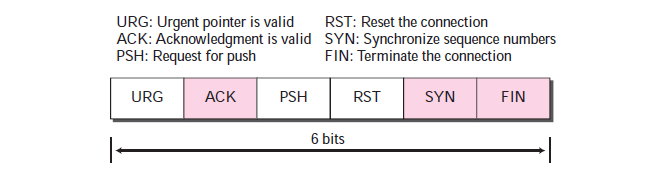


DNS报文格式。

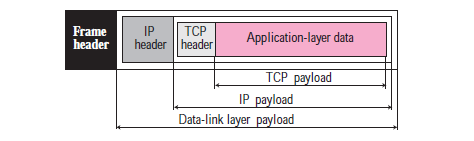




TCP报头格式



TCP flag



网络报文

构造TCP server数据响应报文（已建立连接）

TCP client的数据请求报文request

TCP server的数据响应报文response

response.tcphdr.src\_port = request.tcphdr.dst\_port;

response.tcphdr.dst\_port = request.tcphdr.src\_port;

response.tcphdr.seq\_no = request.tcphdr.ack\_no;

response.tcphdr.ack\_no = request.tcphdr.seq\_no + request.tcphdr.payload\_len;

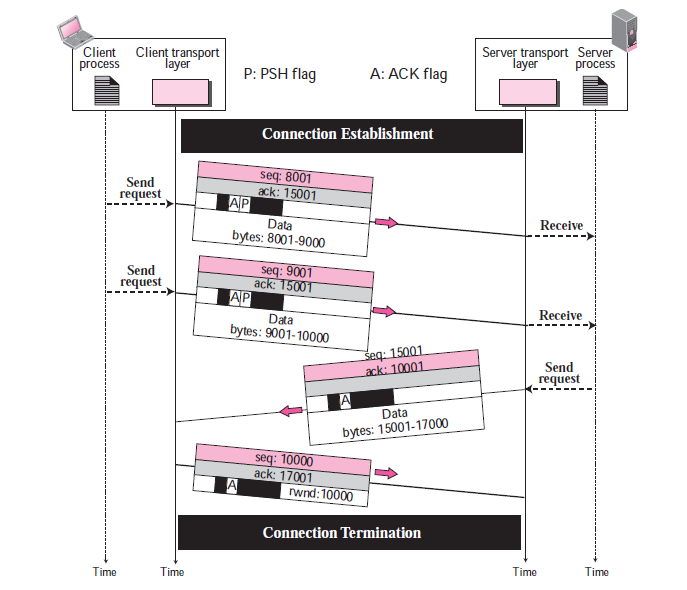
request.tcphdr.payload\_len = request.iphdr.total\_len - request.iphdr.hdr\_len - request.tcphdr.hdr\_len;

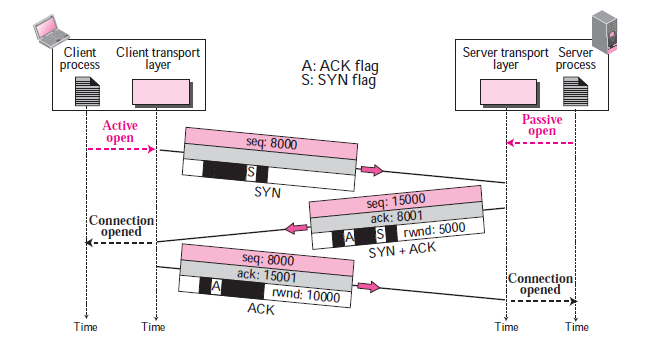
response.tcphdr.flag = TH\_ACK;

（RST报文的话response.tcphdr.flag = TH\_RST）;

（如果响应报文带数据并且需要快速推送到应用层的话，response.tcphdr.flag |= TH\_PSH）;

（如果响应报文是最后一个报文，即服务端打算主动关闭连接的话，response.tcphdr.flag |= TH\_FIN）；





构造TCP server建立连接握手响应报文（第二个报文）

response.tcphdr.src\_port = request.tcphdr.dst\_port;

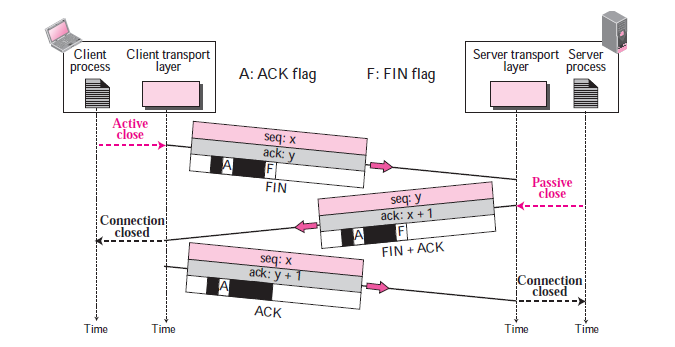
response.tcphdr.dst\_port = request.tcphdr.src\_port;

response.tcphdr.seq\_no = any;

response.tcphdr.ack\_no = request.tcphdr.seq\_no + 1;

response.tcphdr.flag = TH\_SYN | TH\_ACK;

（RST报文的话response.tcphdr.flag = TH\_RST | TH\_ACK）;



构造TCP server关闭连接握手响应报文（第二个报文）

response.tcphdr.src\_port = request.tcphdr.dst\_port;

response.tcphdr.dst\_port = request.tcphdr.src\_port;

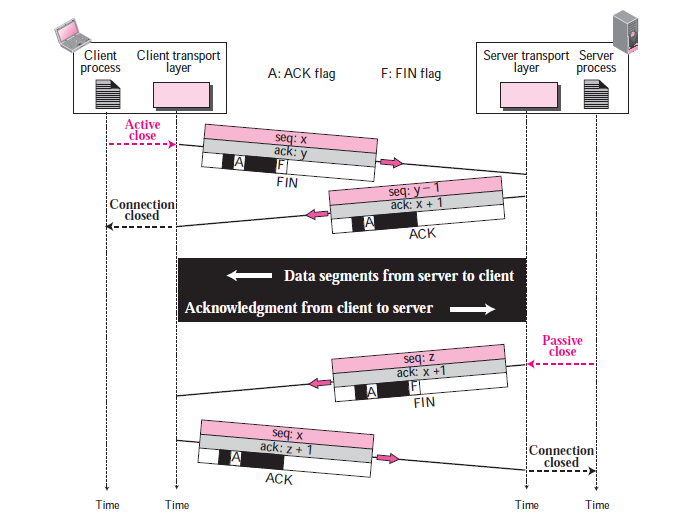
response.tcphdr.seq\_no = request.tcphdr.ack\_no;

response.tcphdr.ack\_no = request.tcphdr.seq\_no + 1;

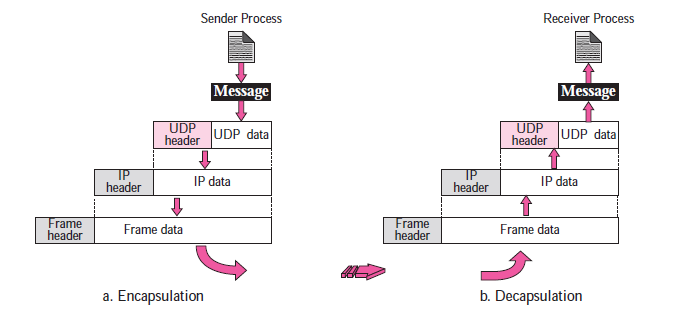
response.tcphdr.flag = TH\_FIN | TH\_ACK;

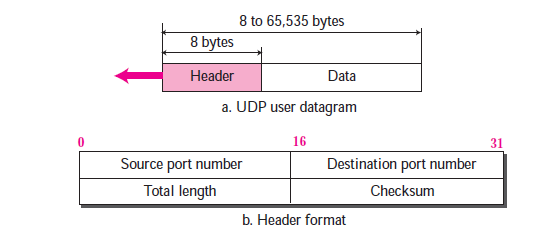
（RST报文的话response.tcphdr.flag = TH\_RST | TH\_ACK）;

半关闭连接



UDP打包和解包过程





UDP报头格式

构造UDP server数据响应报文

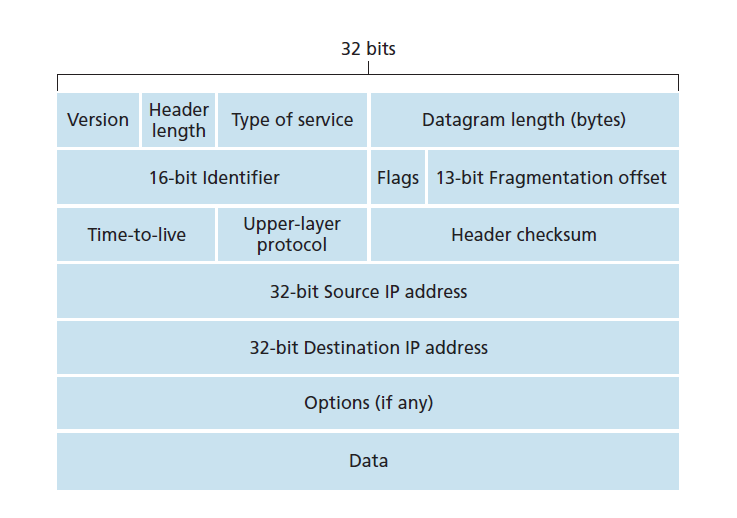
UDP client的数据请求报文request

UDP server的数据响应报文response

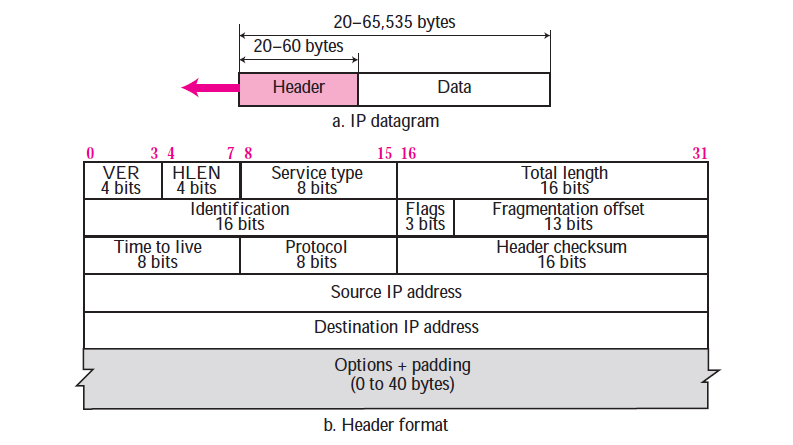
response.udphdr.src\_port = request.udphdr.dst\_port;

response.udphdr.dst\_port = request.udphdr.src\_port;

response.udphdr.total\_len = response.udphdr.payload\_len + 8;



IP报文格式



IP报头格式

构造IP server数据响应报文

IP client的数据请求报文request

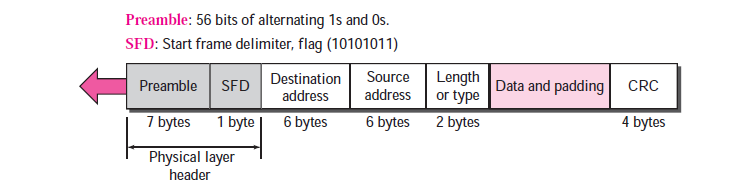
IP server的数据响应报文response

response.iphdr.src\_addr = request.iphdr.dst\_addr;

response.iphdr.dst\_addr = request.iphdr.src\_addr;

response.iphdr.total\_len = response.iphdr.hdr\_len + response.transport.hdr\_len + response.application.payload\_len;

response.iphdr.protocol = request.iphdr.protocol;



以太网帧格式

构造Ethernet server数据响应报文

Ethernet client的数据请求报文request

Ethernet server的数据响应报文response

response.ethhdr.src\_mac = request.ethhdr.dst\_mac;

response.ethhdr.dst\_mac = request.ethhdr.src\_mac;

response.ethhdr.payload\_len = response.iphdr.total\_len;

response.ethhdr.ether\_type= request.ethhdr. ether\_type;