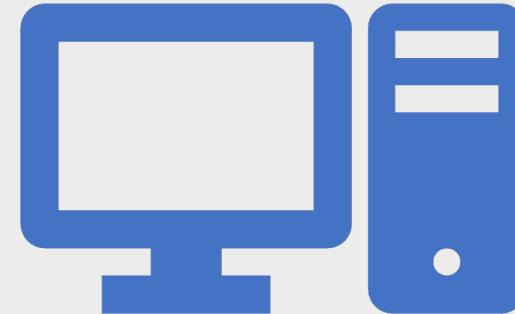
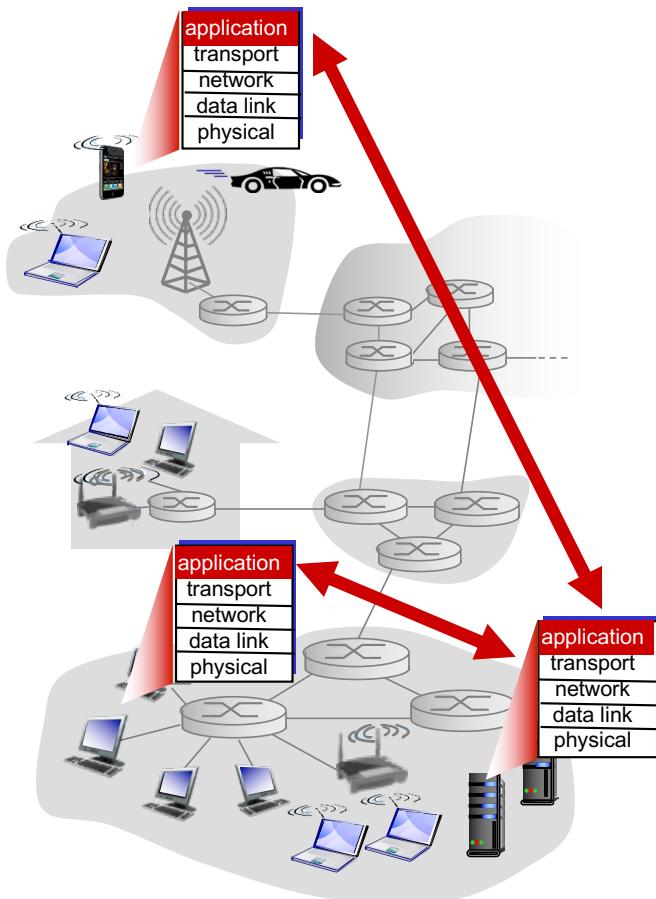


Lecture 2 – Application Layer (1)

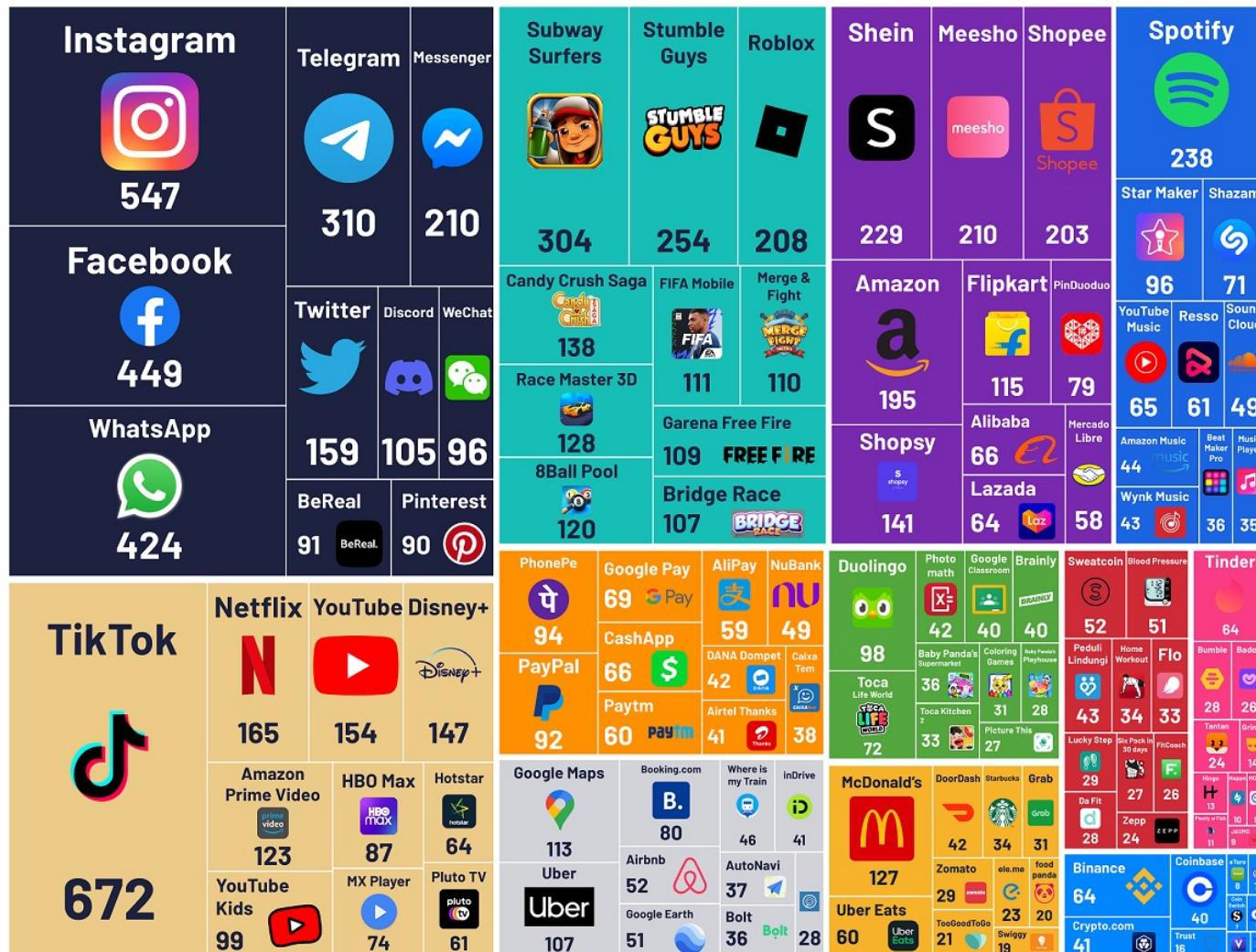
- 1. Introduction to Application Layer**
- 2. Principles of Network Applications**
- 3. Web Application - HTTP**



APP layer in TCP/IP stack

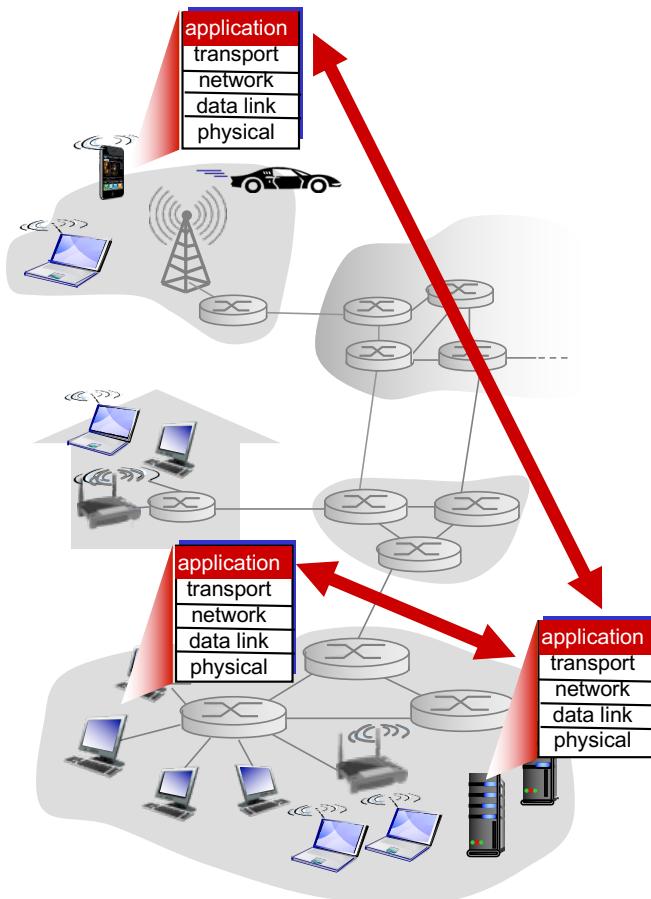


- Top of the TCP/IP stack
- Closest to the end-user
- Provides protocols for TX/RX over the network 提供在网格上发送/接收的协议
- Rich network applications



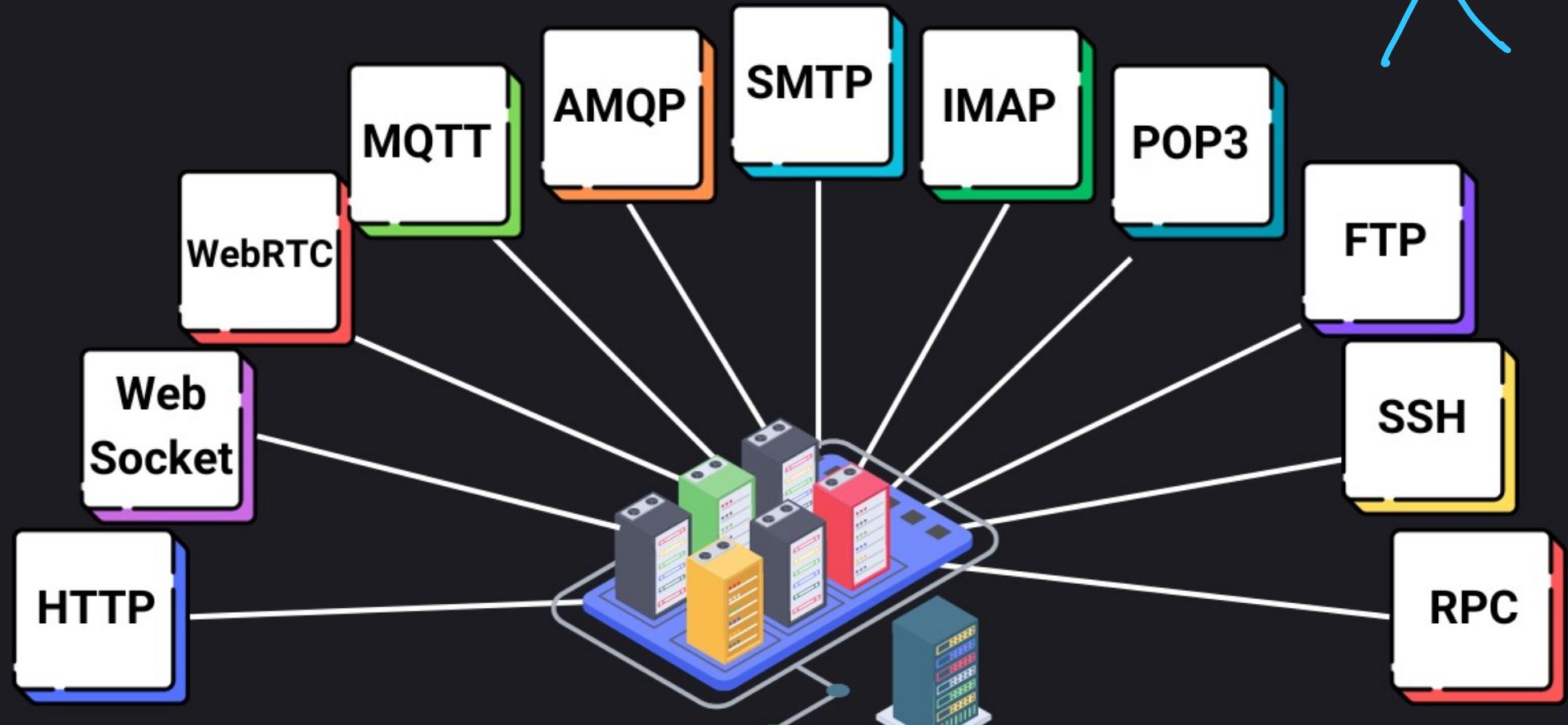
<https://www.businessoffapps.com/data/app-reports/>

APP layer in TCP/IP stack

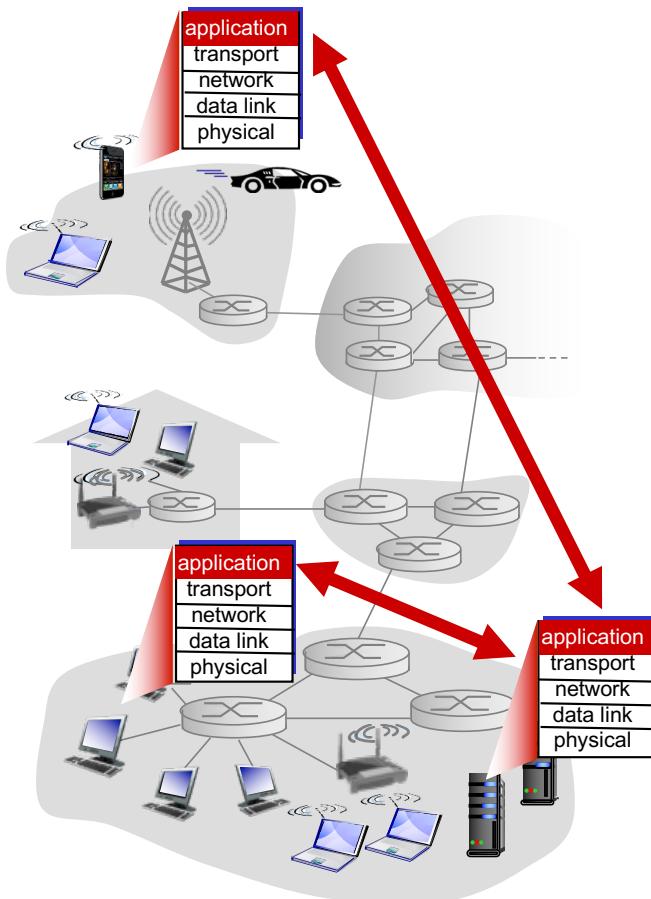


- Top of the TCP/IP stack
- Closest to the end-user
- Provides protocols for TX/RX over the network
- Rich network applications
- Many protocols for application layers

APPLICATION PROTOCOLS



APP layer in TCP/IP stack



- Top of the TCP/IP stack
- Closest to the end-user
- Provides protocols for TX/RX over the network
- Rich network applications
- Many protocols for application layers
- Can be customized easily! 可轻松定制使用



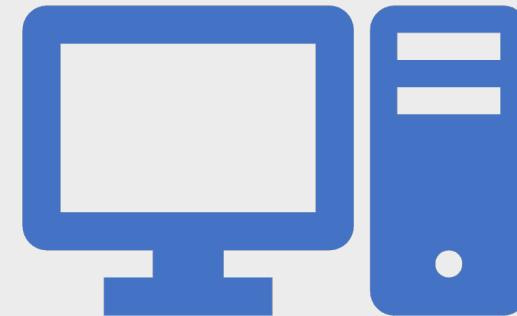
OR



openWRT

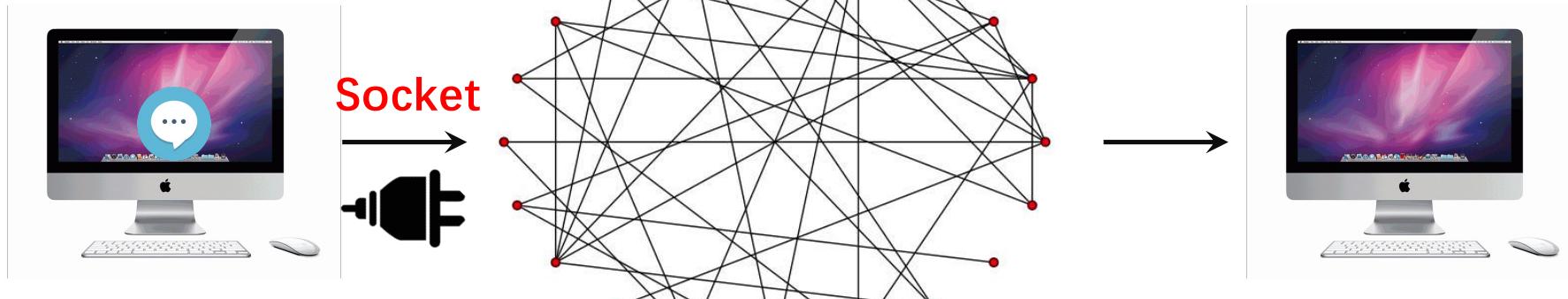
Lecture 2 – Application Layer (1)

- 1. Introduction to Application Layer**
- 2. Principles of Network Applications**
- 3. Web Application - HTTP**

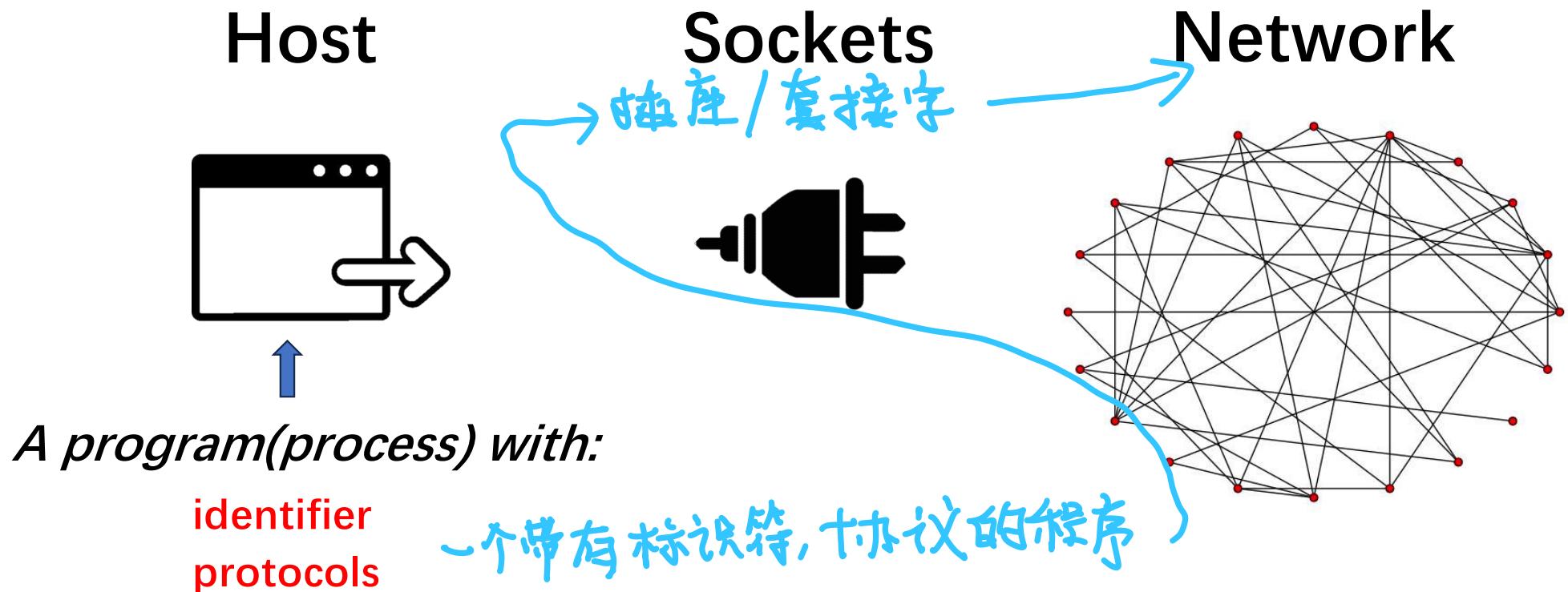




What do we use to deliver messages?

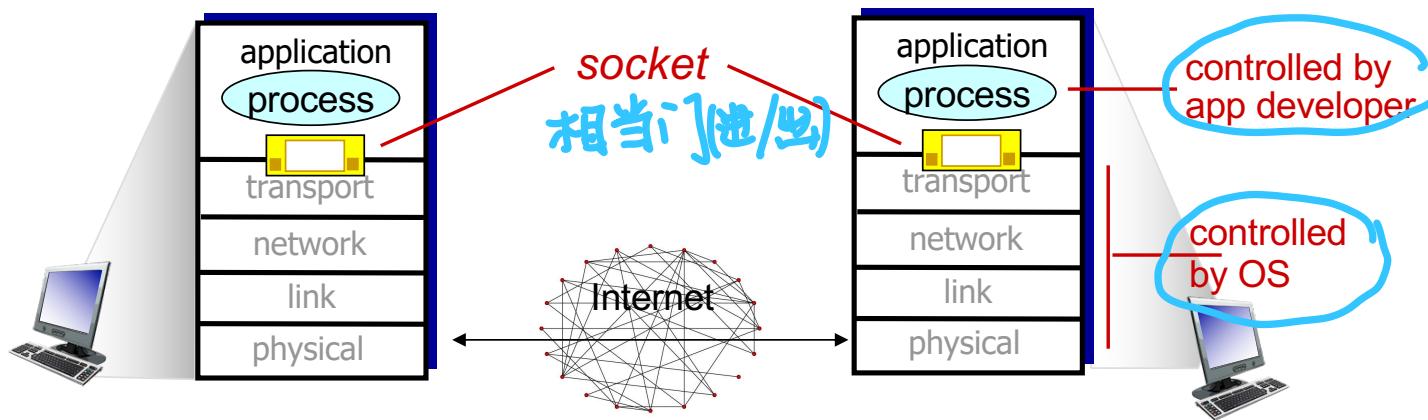


What do we use to deliver messages?



Sockets

- Process sends/receives messages to/from its socket
- Socket analogous to door 将socket类比为door
 - Sending process "push" message out of the door
 - Sending process relies on transport infrastructure on other side of door to deliver message to socket at receiving process



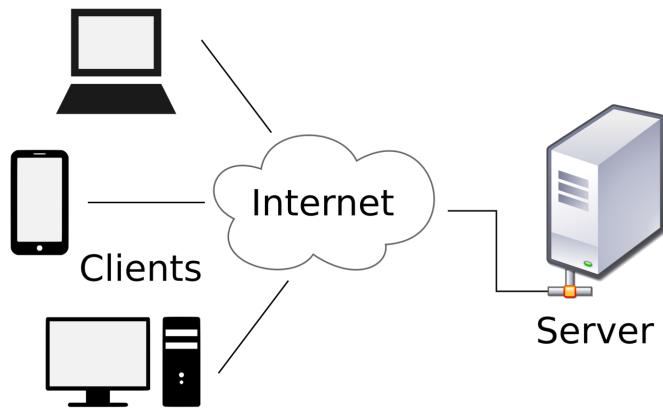
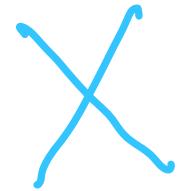
Addressing processes

- To receive messages, process must have *identifier*
- Host device has unique 32-bit IPv4 and/or 128-bit IPv6
- Process network identifier:
 - IPv4:port 192.168.1.100:80
 - [IPv6]:port [240e:3a1:4cb1:69d0:f40c:4269:74a2:7ea3]:80

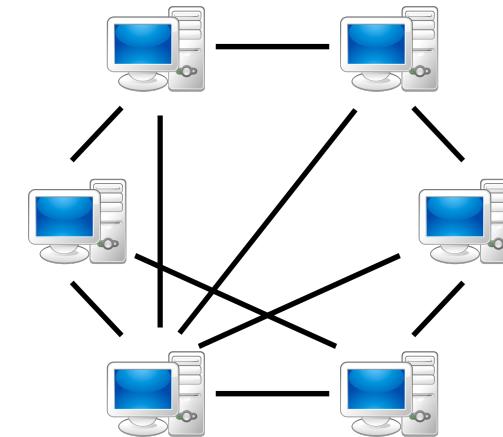
端口号



Architectures for applications



Client-server



Peer-to-peer

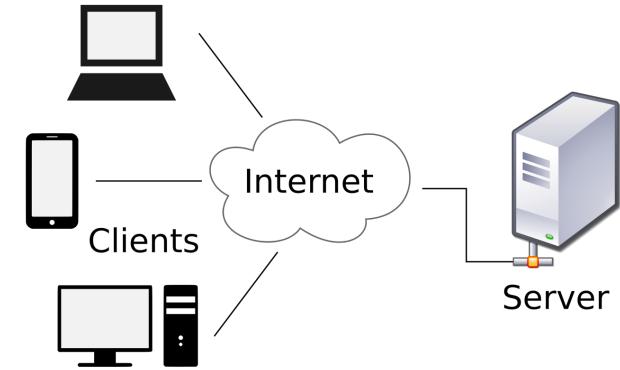
The background of the slide is a photograph of a server room. Rows of server racks are visible, each with numerous small, glowing blue and green lights. The floor is a polished grey, and a set of double doors is visible at the end of the aisle. The overall atmosphere is dark and technical.

Think
How to distinguish server and client?

Client-server architecture

- **Server**

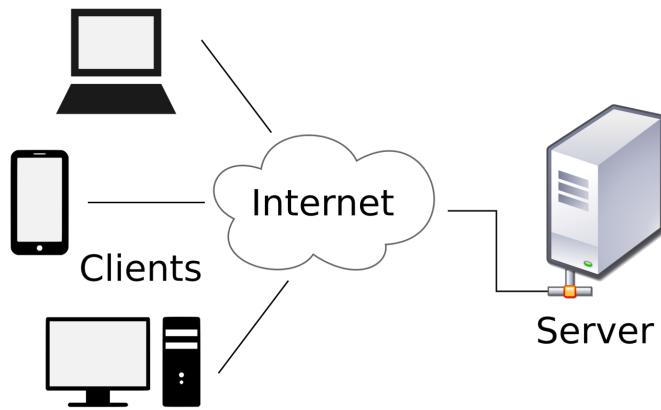
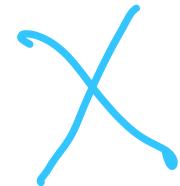
- Always-on host **一直在线**
- Permanent IP address **固定IP地址**
- High performance / Distributed computing
- Server process: waits to be contacted (Listen)



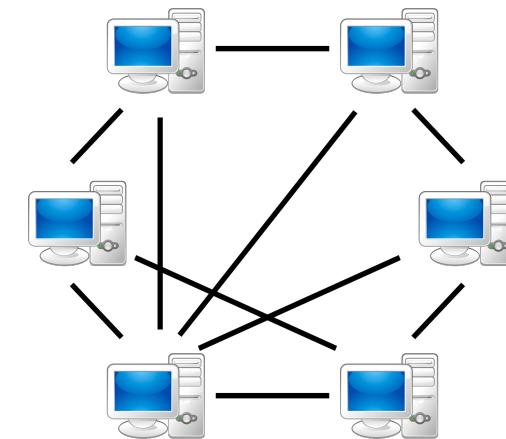
- **Clients**

- Link to the server for service
- May be intermittently connect to the internet
- Dynamic IP address
- Do not communicate directly with each other
- Client process: initiates communication **客户端进程发起通信**

Architectures for applications



Client-server



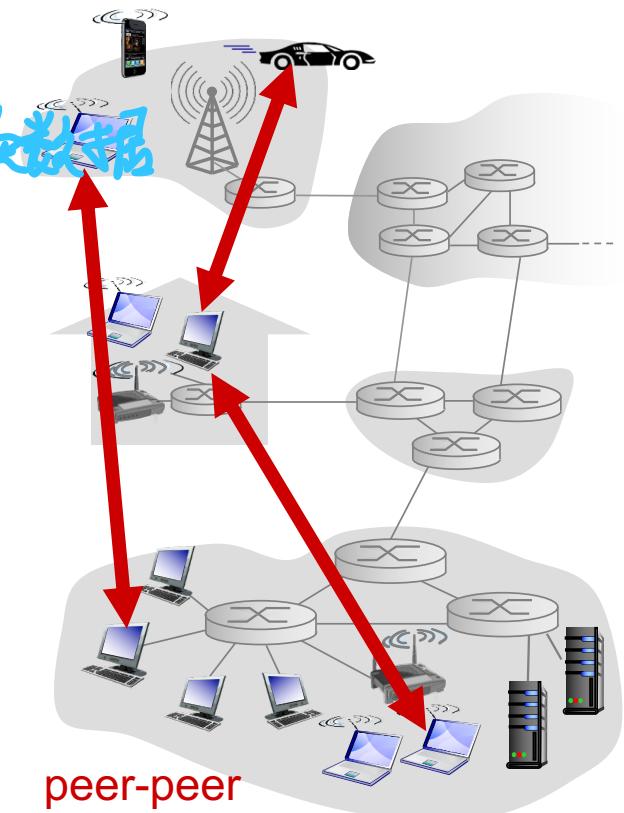
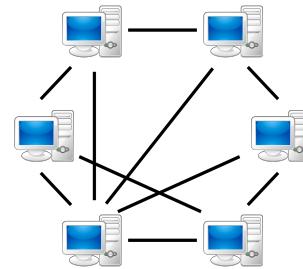
Peer-to-peer

(peer to peer)

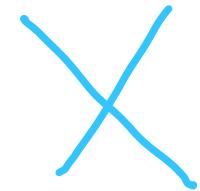
P2P architecture

- No always-on server is needed
- End systems directly exchange data
終端系統直接交換數據
- Client process / server process on the same host
- Peers request service from other peers, provide service in return to other peers
 - *Self scalability – new peers bring new service capacity, as well as new service demands*
- Peers are intermittently connected *間歇性連接*
- Dynamic IP addresses
- Q: Did you use Thunder Downloader(迅雷)?
Why does it download so fast? *P2P*

用户可自行决定是否提供向他人共享



Try to have your cloud server (optional)



- Most cloud providers provide student discount.
- Ali Cloud, Tencent Cloud...

App-layer protocol defines

- **Architecture:** CS or/and P2P
- **Types of messages exchanged**
 - e.g., request, response
- **Message syntax:** 消息中字段以及如何划分
 - what fields in messages & how fields are delineated
- **Message semantics:** 消息字段中信息的含义
 - meaning of information in fields
- **Message timing:** when and how 消息何时以及如何发送

Open protocols:

- Defined in RFCs (Request for Comments)
- Allows for interoperability 互操作性
- e.g., HTTP, SMTP, FTP

Private protocols:

- e.g., Skype, Games, your own protocols...



What kinds of transport service do we need?

Transport service requirements

Data Integrity

- 100% reliable data transfer or
- Tolerate some loss

Timing

- Some apps require low delay

Throughput

- Some apps require minimum amount of throughput to be “effective”
(高吞吐量
才有效率，
网游)
- Others do not require

- Some apps require encryption
- Data integrity check

Security

Transport service requirements: common apps

Applications	Data loss	Throughput	Time sensitive
File transfer	No loss 不允许丢失	Elastic 弹性	No
E-mail	No loss	Elastic	No
Web documents	No loss	Elastic	No
Real-time video/audio	Loss-tolerant 允许部分丢失	Based on quality* 有要求最低不低于多少	Yes 100 ms
Stored video/audio	Loss-tolerant	Based on quality*	Yes few second
Interactive games	Loss-tolerant	Few kpbs	Yes 100 ms
Text messaging	No loss	Elastic	Yes or No

* Audio: 5k to 1Mpbs, Video: 10kpbs - 10 Mbps or more



23

Internet transport protocols services



TCP ->



<- UDP

★这是传输层的协议

Internet transport protocols services

TCP service:

- **Reliable transport** between sending and receiving process
- **Flow control:** sender won't overwhelm receiver
发送方不能使接收方过载
- **Congestion control:** throttle sender when network overloaded
发现网络拥塞,减少发送方数据量,避免加剧
- **Does not offer:** timing, minimum throughput guarantee, security
本身不提供 定时
- **Connection-oriented:** setup required between client and server processes
需要先建立连接

Internet transport protocols services

UDP service:

- **Unreliable data transfer** between sending and receiving process
- **Does not offer:** reliability, flow control, congestion control, timing, throughput guarantee, security, or connection setup,
- Looks so bad... But...

Why do we still need UDP?

Simple and fast!

Application vs transport protocols

Application	App layer protocol	Underlying transport Protocol
E-mail	SMTP [RFC 2821], POP3, IMAP	TCP
Remote terminal Access	Telnet [RFC 854], SSH	TCP
Web	HTTP [RFC 2616] HTTPS	TCP
File Transfer	FTP [RFC 959], SFTP	TCP
Multimedia	HTTP / RTP [RFC 1889]	TCP or UDP
VoIP 网络语音通话	SIP, RTP or proprietary	TCP or UDP

语音/视频/文件发起协议 → 实时传输控制协议

Securing TCP - Secure Sockets Layer - SSL

①

TCP & UDP

- No encryption
- Cleartext psws -> Internet

本身不提供加密

③

SSL is at app layer

- Apps use SSL libraries, that "talk" to TCP

②

SSL

- Provides encrypted TCP connection
- Data integrity
- End-point authentication

④

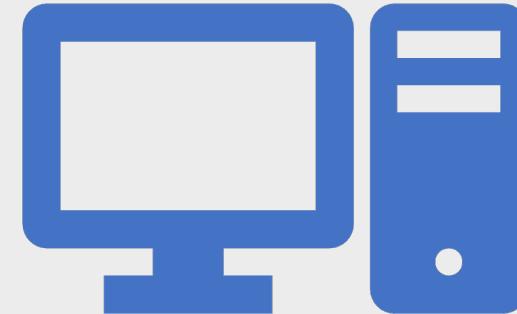
SSL socket API

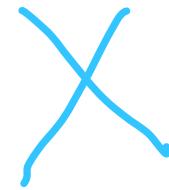
- Cleartext psw -> encrypted psw -> Internet
- Lecture 11/12 will talk more

加密密码

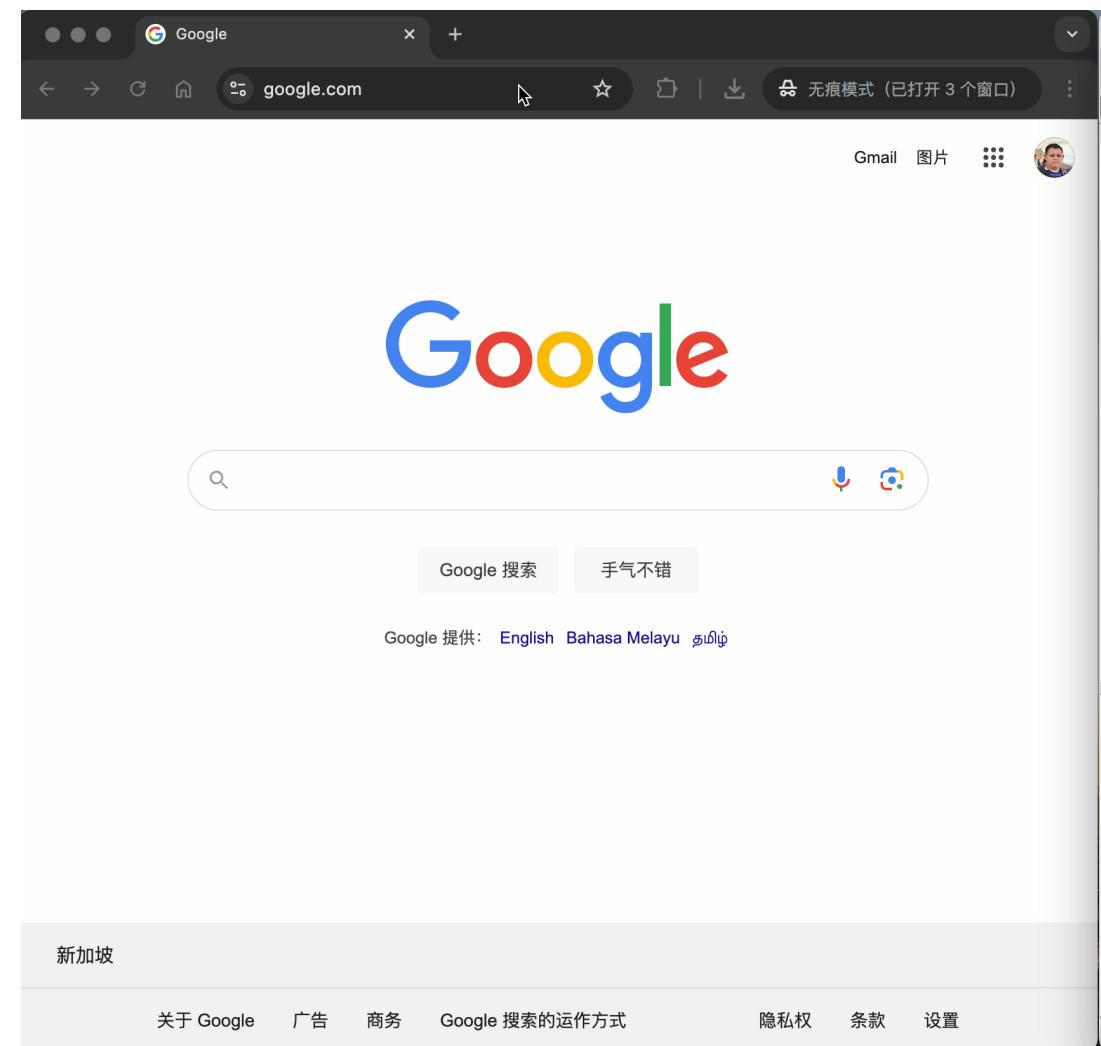
Lecture 2 – Application Layer (1)

1. Introduction to Application Layer
2. Principles of Network Applications
3. Web Application - HTTP

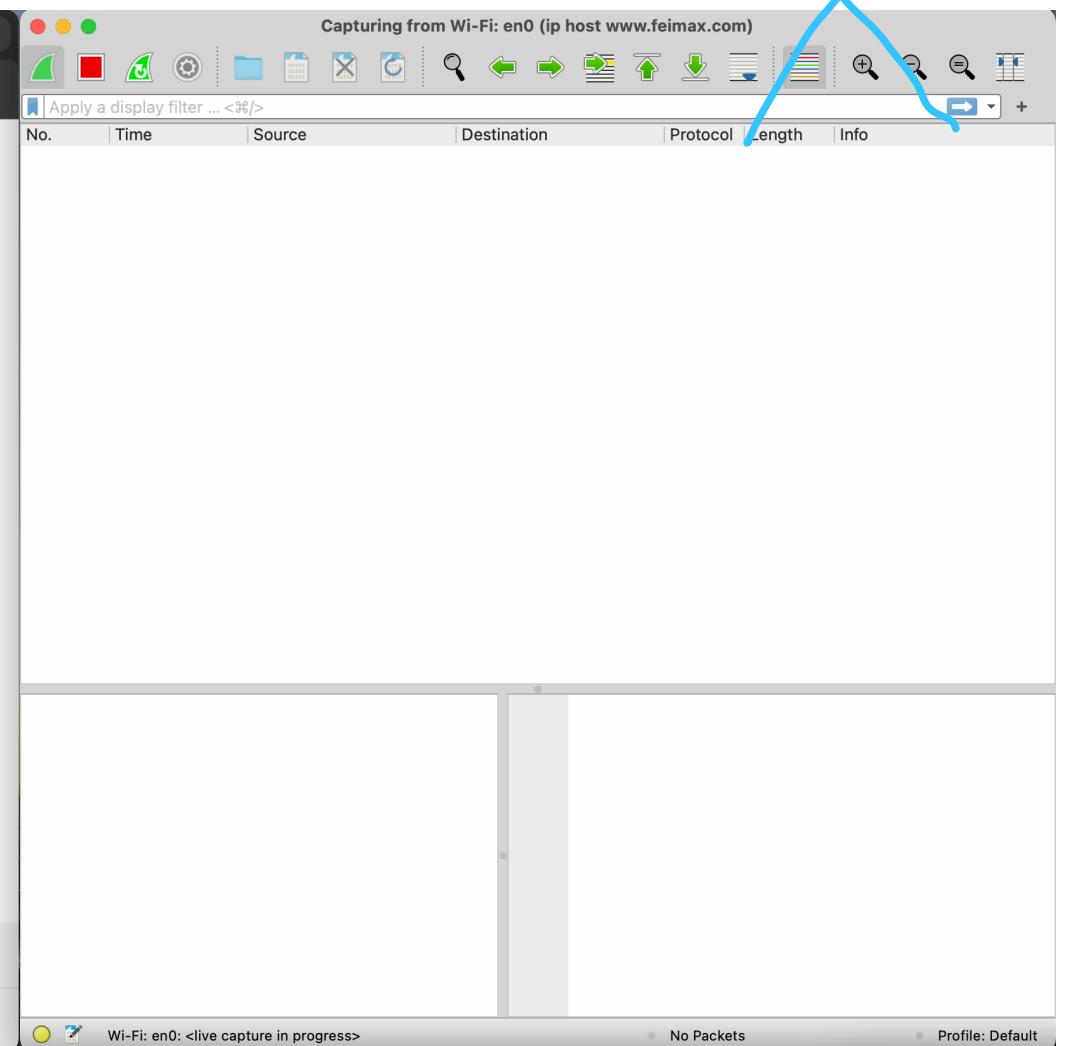




HTTP = WEBPAGE?

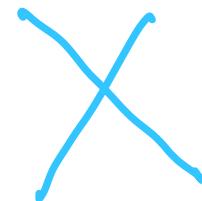


Chrome



Wireshark

32



Feimax's Homepage

△ 不安全 feimax.com

Welcome to Feimax's page—a simple and pure HTML site.

Welcome to my little corner of the world! Here, I share snippets of my life and heartfelt reflections.

This is a photo I took yesterday (2024-09-17) of a very big moon:



Last night's moon was exceptionally bright, and I captured its beauty with my camera. I hope you can also feel this tranquility and wonder.

Aside from photography, I love traveling and reading. In the days to come, I'll be sharing more photos, travel stories, and book insights here.

Thank you for visiting, and I wish you a wonderful day!

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Feimax's Homepage</title>
</head>
<body>
    Welcome to Feimax's page—a simple and pure HTML site.<br><br>

    Welcome to my little corner of the world! Here, I share snippets of my life and hear

    This is a photo I took yesterday (2024-09-17) of a very big moon:<br><br>

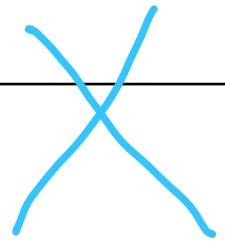
    <img src='20240917.jpg' style="width: 500px;" />
</br>
    Last night's moon was exceptionally bright, and I captured its beauty with my camera

    Aside from photography, I love traveling and reading. In the days to come, I'll be s

    Thank you for visiting, and I wish you a wonderful day!
</body>
```

HTML code (response)

Page



Feimax's Homepage

△ 不安全 feimax.com

Welcome to Feimax's page—a simple and pure HTML site.

Welcome to my little corner of the world! Here, I share snippets of my life and heartfelt reflections.

This is a photo I took yesterday (2024-09-17) of a very big moon:



Last night's moon was exceptionally bright, and I captured its beauty with my camera. I hope you can also feel this tranquility and wonder.

Aside from photography, I love traveling and reading. In the days to come, I'll be sharing more photos, travel stories, and book insights here.

Thank you for visiting, and I wish you a wonderful day!

Page

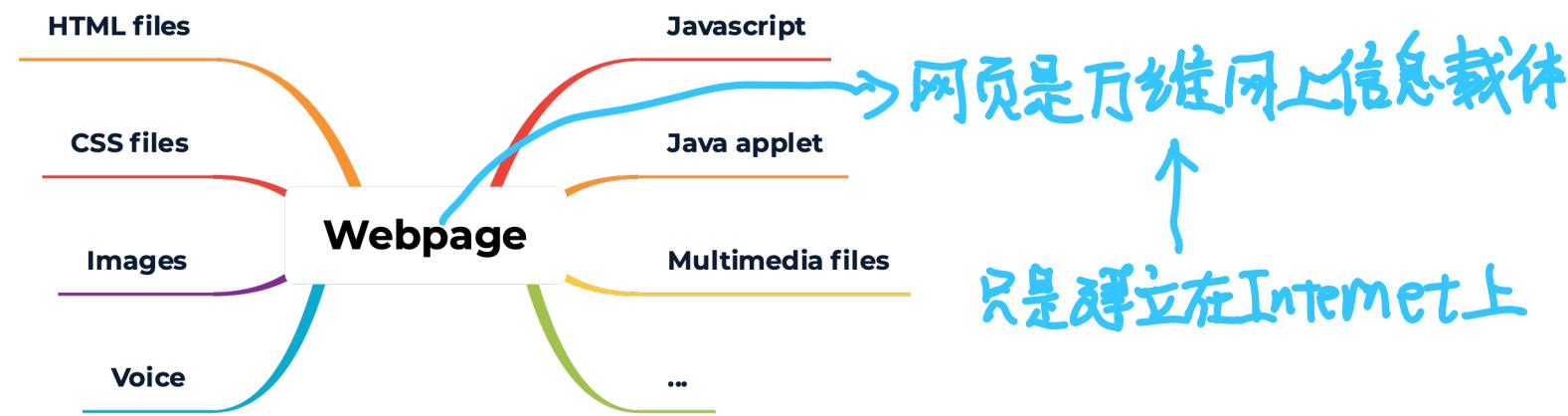
<http://www.feimax.com/20240917.jpg>

HTML code (response)

Web, HTTP and WWW

- WWW: World Wide Web 万维网
- HTTP: Hypertext Transfer Protocol 超文本传输协议

Web = html | 组成
(包含多个引用对象, 通过URL访问)



- Web page consists of *base HTML-file* which includes *several referenced objects, addressable by a URL (uniform resource locator)*

Features of HTTP (1.1)

HTTP: hypertext transfer protocol

(超文本传输协议)

- Application layer protocol
- Client/server model
 - *Client:* browser that requests, receives, (using HTTP protocol) and show Web objects (Render)
 - *Server:* Web server sends (using HTTP protocol) objects in response to requests



Features of HTTP (1.1)

Uses TCP:

1. Client initiates TCP connection (creates socket) to server, port 80(443 for https)
2. Server accepts TCP connection from client
3. HTTP messages (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server)
4. TCP connection closed

http

HTTP is “*无状态的*”

- server maintains no information about past client requests

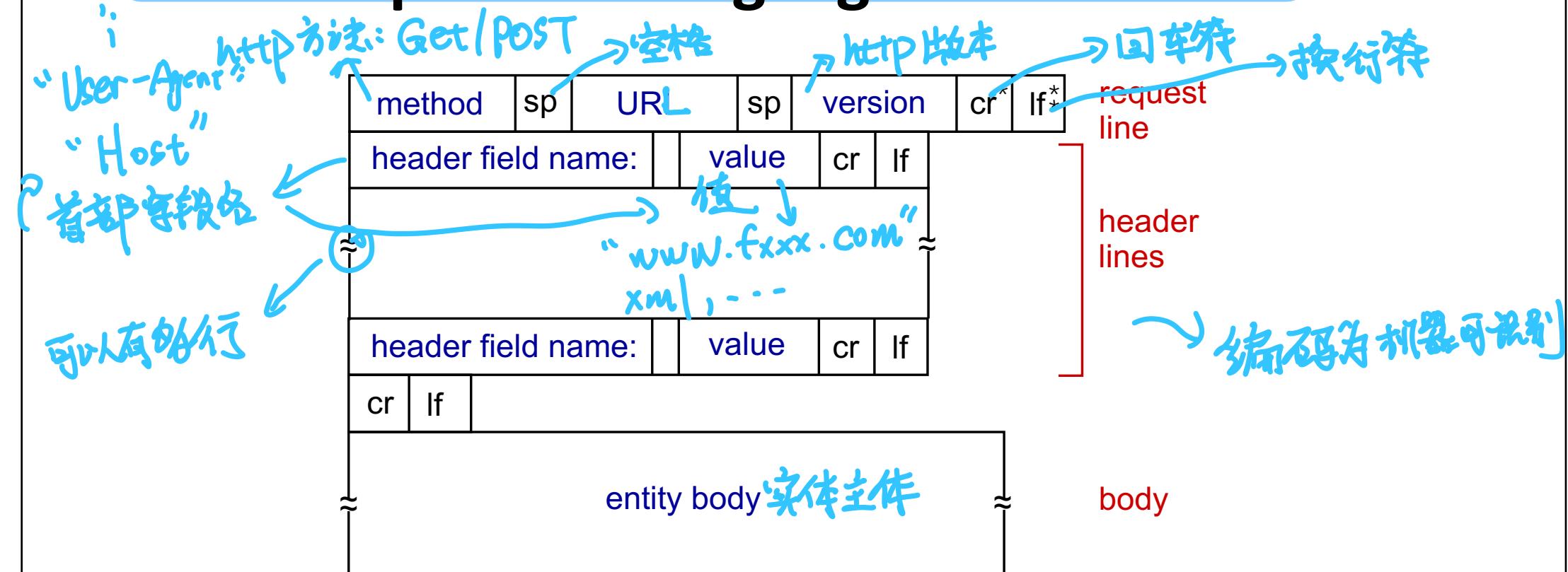
服务器不保存过去客户端请求信息

aside

Protocols that maintain “state” are complex!

- past history (state) must be maintained
- if server/client crashes, their views of “state” may be inconsistent, must be reconciled (保持一致)

1 HTTP request message: general format



*carriage return character 回车

** line-feed character 换行

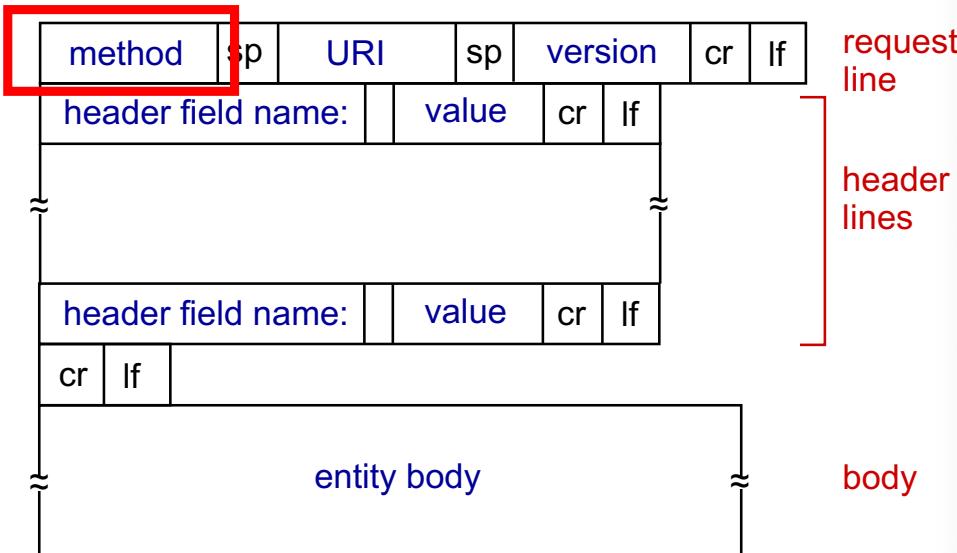
Note: No IP address is included in the HTTP request. Why?





https://www.youtube.com/watch?v=M9RunwrxStw&ab_channel=DR.TYPERWRITER-VenneburgTypewriters 40

HTTP request message: general format



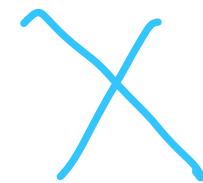
Wireshark · Packet 20 · Wi-Fi: en0 (ip host www.feimax.com)

```

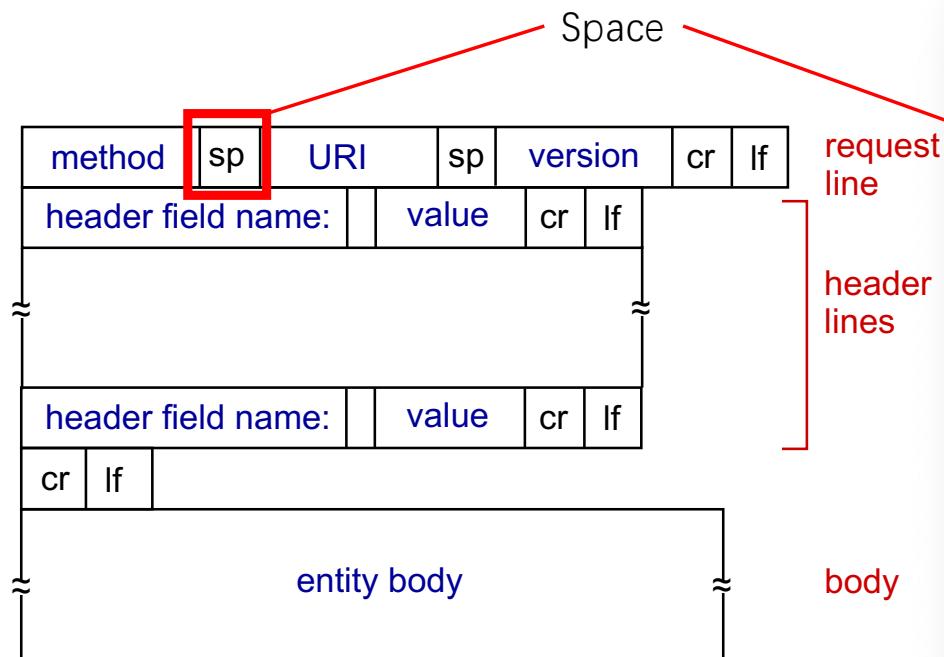
> Frame 20: 489 bytes on wire (3912 bits), 489 bytes captured (3912 bits) on interface
> Ethernet II, Src: Apple_74:26:5d (f4:d4:88:74:26:5d), Dst: Tp-LinkT_0b:d8:5c (00:0c:29:0b:d8:5c)
> Internet Protocol Version 4, Src: 192.168.0.24, Dst: 106.14.20.246
> Transmission Control Protocol, Src Port: 59244, Dst Port: 80, Seq: 1, Ack: 1,
< Hypertext Transfer Protocol
  GET / HTTP/1.1\r\n
  Host: www.feimax.com\r\n
  Connection: keep-alive\r\n
  Upgrade-Insecure-Requests: 1\r\n
  User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.122 Safari/537.36
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
  Accept-Encoding: gzip, deflate\r\n
  Accept-Language: zh\r\n
  \r\n
  [Full request URI: http://www.feimax.com/]
  [HTTP request 1/2]
  [Response in frame: 22]
  [Next request in frame: 24]
```

Show packet bytes

Help



HTTP request message: general format



Wireshark · Packet 20 · Wi-Fi: en0 (ip host www.feimax.com)

```

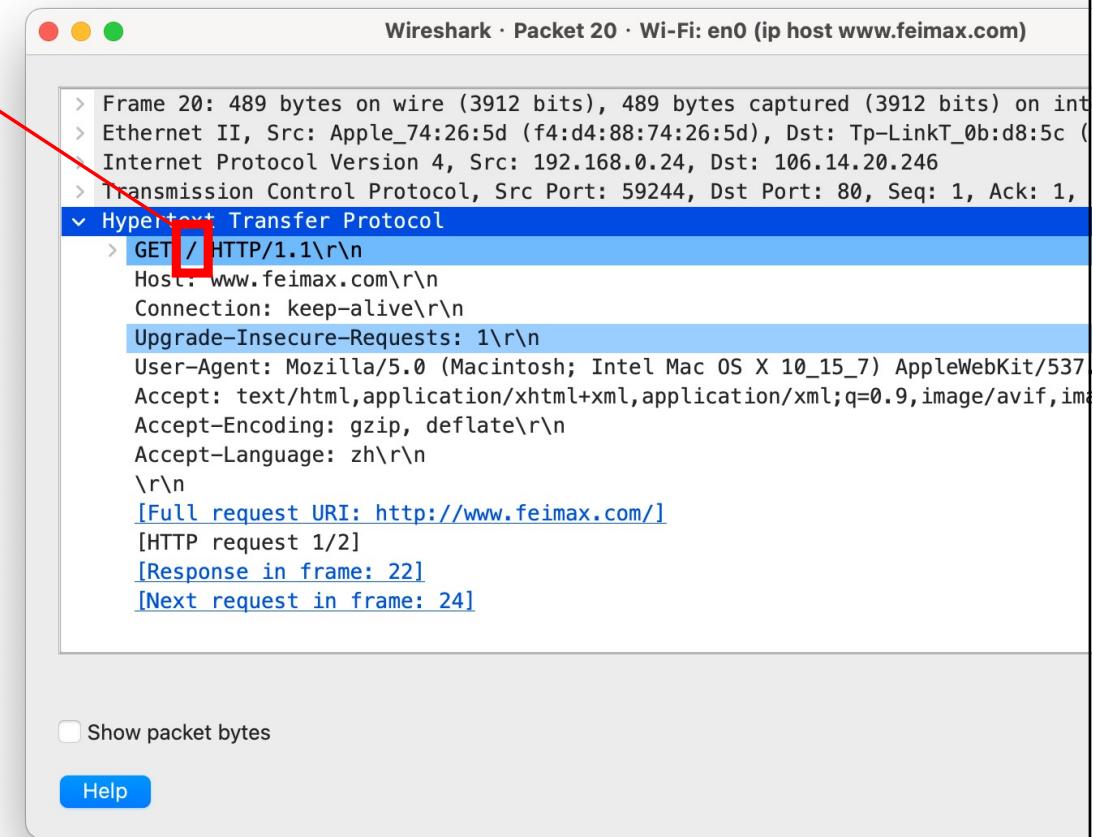
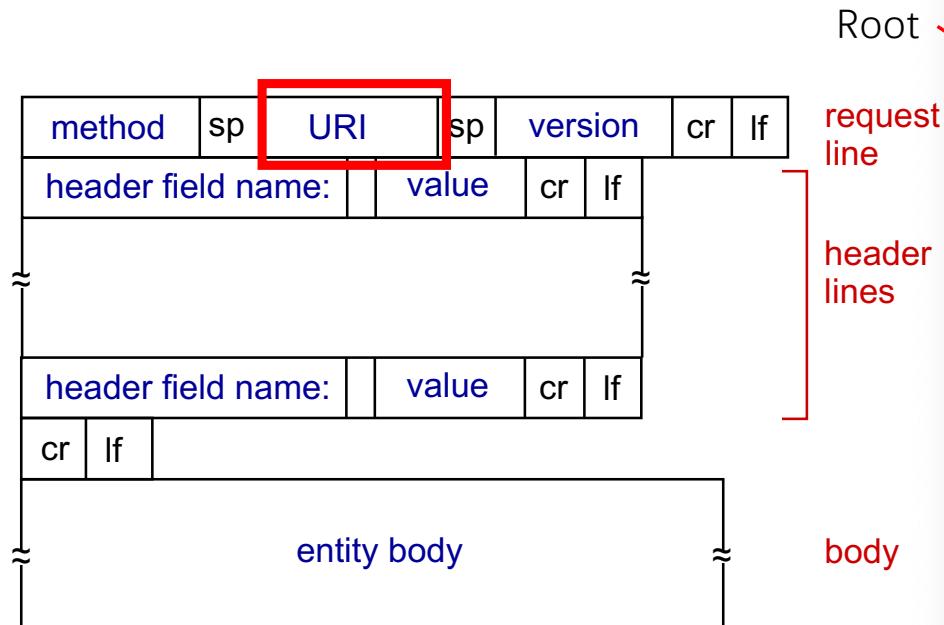
> Frame 20: 489 bytes on wire (3912 bits), 489 bytes captured (3912 bits) on interface en0
> Ethernet II, Src: Apple_74:26:5d (f4:d4:88:74:26:5d), Dst: Tp-LinkT_0b:d8:5c (00:0c:29:0b:d8:5c)
> Internet Protocol Version 4, Src: 192.168.0.24, Dst: 106.14.20.246
> Transmission Control Protocol, Src Port: 59244, Dst Port: 80, Seq: 1, Ack: 1, Len: 489
< HTML Hypertext Transfer Protocol
> GET / HTTP/1.1\r\n
Host: www.feimax.com\r\n
Connection: keep-alive\r\n
Upgrade-Insecure-Requests: 1\r\n
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.122 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Encoding: gzip, deflate\r\n
Accept-Language: zh\r\n
\r\n
[Full request URI: http://www.feimax.com/]
[HTTP request 1/2]
[Response in frame: 22]
[Next request in frame: 24]
```

Show packet bytes

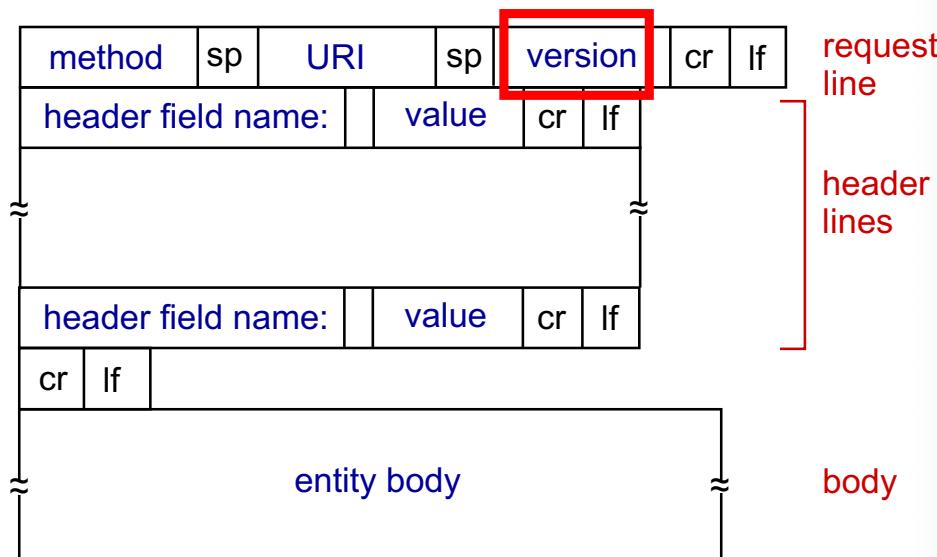
Help

HTTP request message: general format

X



HTTP request message: general format



Wireshark · Packet 20 · Wi-Fi: en0 (ip host www.feimax.com)

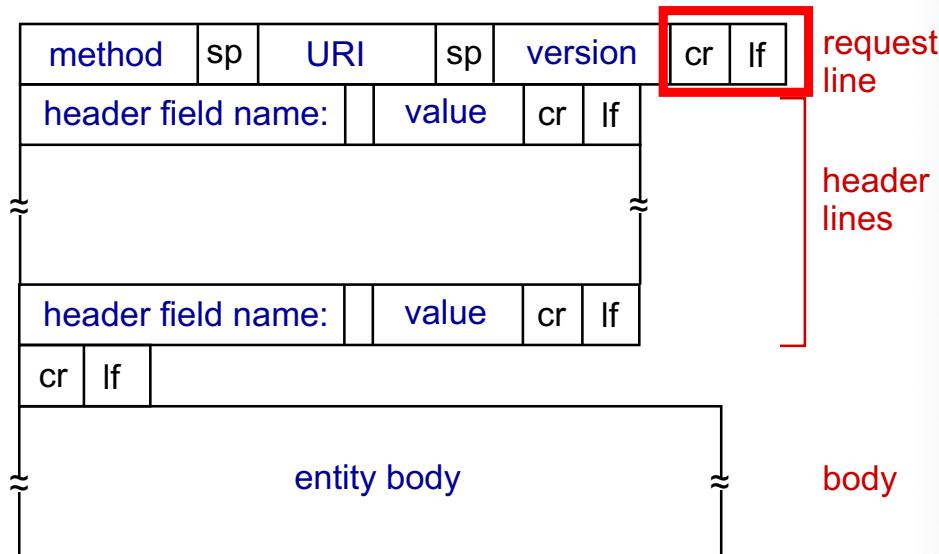
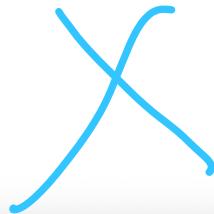
```

> Frame 20: 489 bytes on wire (3912 bits), 489 bytes captured (3912 bits) on interface en0
> Ethernet II, Src: Apple_74:26:5d (f4:d4:88:74:26:5d), Dst: Tp-LinkT_0b:d8:5c (00:0c:29:0b:d8:5c)
> Internet Protocol Version 4, Src: 192.168.0.24, Dst: 106.14.20.246
> Transmission Control Protocol, Src Port: 59244, Dst Port: 80, Seq: 1, Ack: 1,
< Hypertext Transfer Protocol
  > GET / HTTP/1.1\r\n
    Host: www.feimax.com\r\n
    Connection: keep-alive\r\n
    Upgrade-Insecure-Requests: 1\r\n
    User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.122 Safari/537.36
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
    Accept-Encoding: gzip, deflate\r\n
    Accept-Language: zh\r\n
    \r\n
    [Full request URI: http://www.feimax.com/]
    [HTTP request 1/2]
    [Response in frame: 22]
    [Next request in frame: 24]

 Show packet bytes
Help

```

HTTP request message: general format



Wireshark · Packet 20 · Wi-Fi: en0 (ip host www.feimax.com)

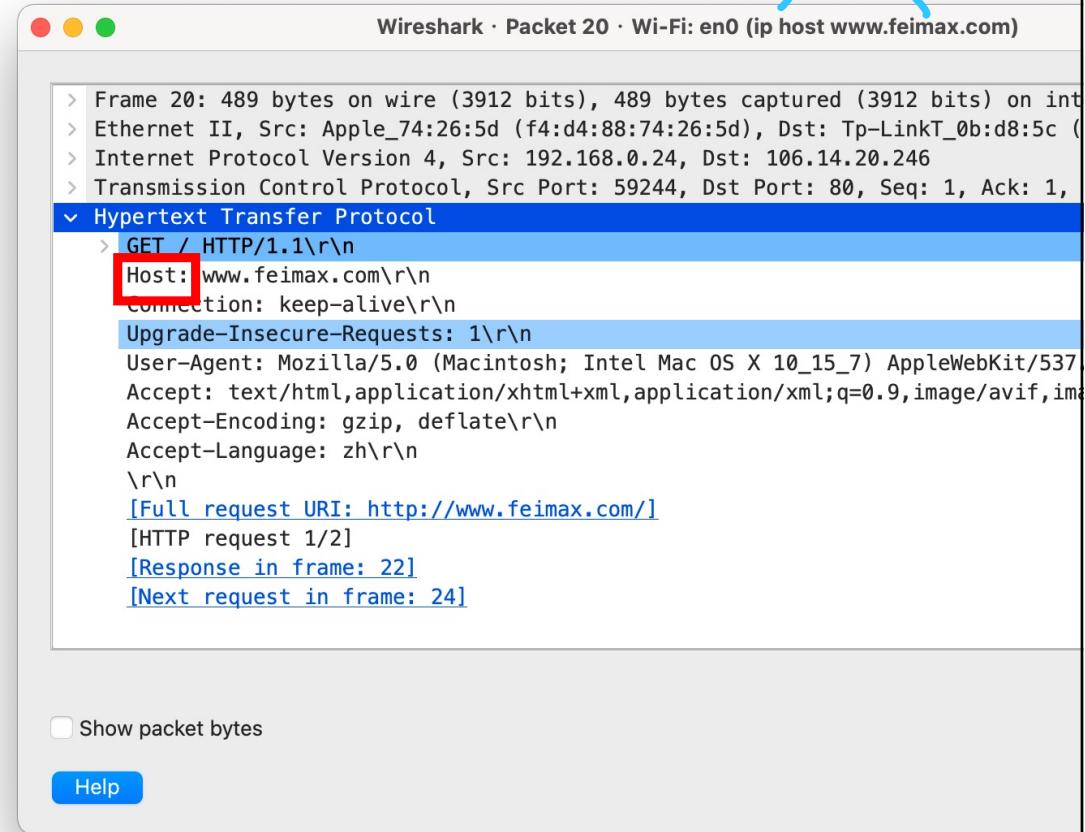
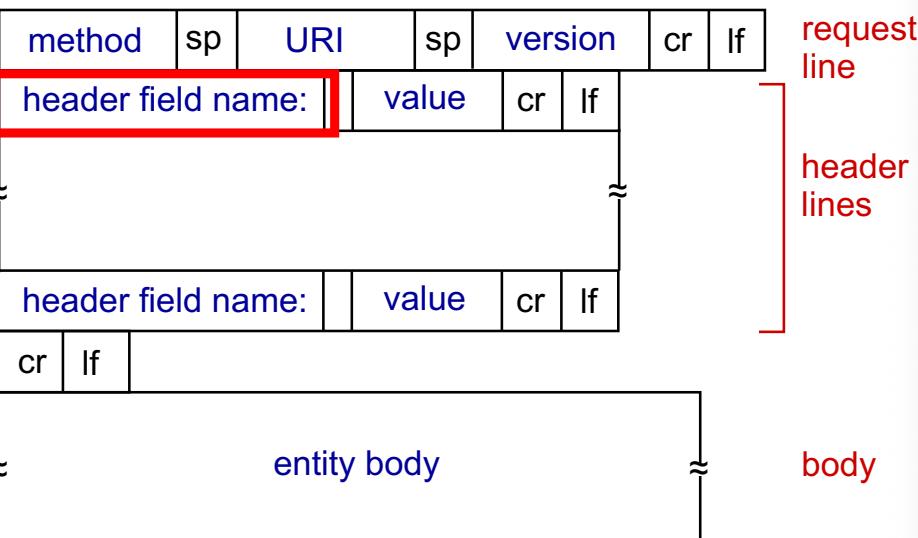
```

> Frame 20: 489 bytes on wire (3912 bits), 489 bytes captured (3912 bits) on interface
> Ethernet II, Src: Apple_74:26:5d (f4:d4:88:74:26:5d), Dst: Tp-LinkT_0b:d8:5c (00:0c:29:0b:d8:5c)
> Internet Protocol Version 4, Src: 192.168.0.24, Dst: 106.14.20.246
> Transmission Control Protocol, Src Port: 59244, Dst Port: 80, Seq: 1, Ack: 1,
< Hypertext Transfer Protocol
  > GET / HTTP/1.1\r\n
    Host: www.feimax.com\r\n
    Connection: keep-alive\r\n
    Upgrade-Insecure-Requests: 1\r\n
    User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.122 Safari/537.36
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
    Accept-Encoding: gzip, deflate\r\n
    Accept-Language: zh\r\n
    \r\n
    [Full request URI: http://www.feimax.com/]
    [HTTP request 1/2]
    [Response in frame: 22]
    [Next request in frame: 24]
```

Show packet bytes

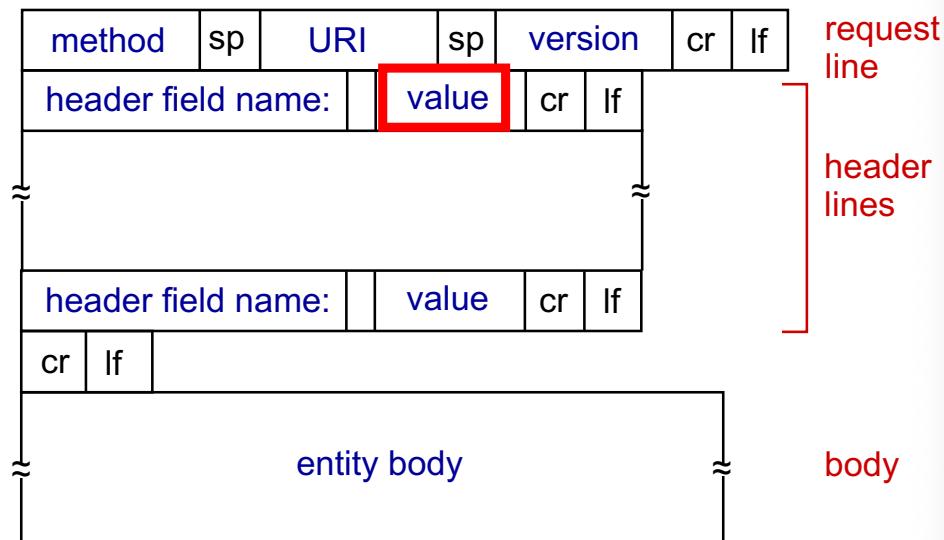
Help

HTTP request message: general format





HTTP request message: general format



Wireshark · Packet 20 · Wi-Fi: en0 (ip host www.feimax.com)

```

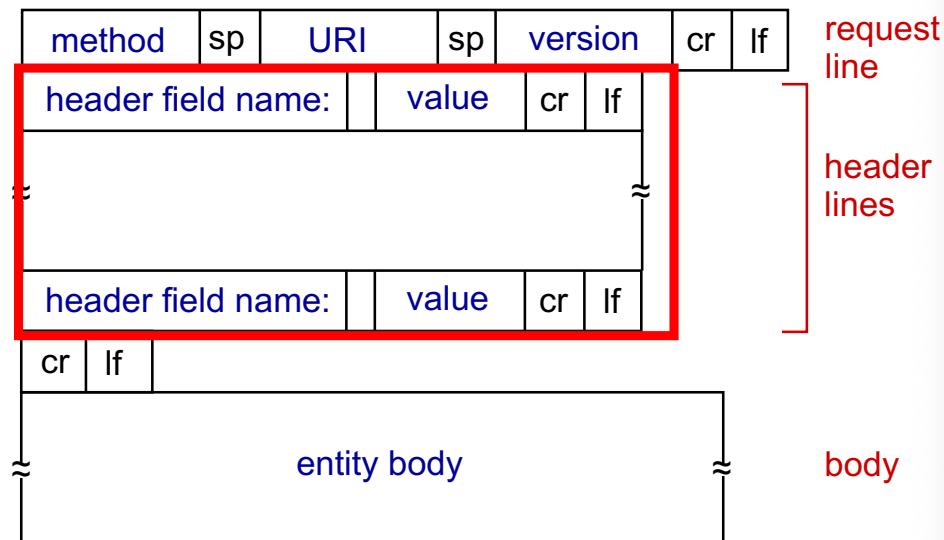
> Frame 20: 489 bytes on wire (3912 bits), 489 bytes captured (3912 bits) on interface en0
> Ethernet II, Src: Apple_74:26:5d (f4:d4:88:74:26:5d), Dst: Tp-LinkT_0b:d8:5c (00:0c:29:0b:d8:5c)
> Internet Protocol Version 4, Src: 192.168.0.24, Dst: 106.14.20.246
> Transmission Control Protocol, Src Port: 59244, Dst Port: 80, Seq: 1, Ack: 1, Len: 489
< Hypertext Transfer Protocol
  > GET / HTTP/1.1\r\n
  Host: www.feimax.com\r\n
  Connection: keep-alive\r\n
  Upgrade-Insecure-Requests: 1\r\n
  User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.122 Safari/537.36
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
  Accept-Encoding: gzip, deflate\r\n
  Accept-Language: zh\r\n
  \r\n
  [Full request URI: http://www.feimax.com/]
  [HTTP request 1/2]
  [Response in frame: 22]
  [Next request in frame: 24]

 Show packet bytes
Help

```



HTTP request message: general format



Wireshark · Packet 20 · Wi-Fi: en0 (ip host www.feimax.com)

```

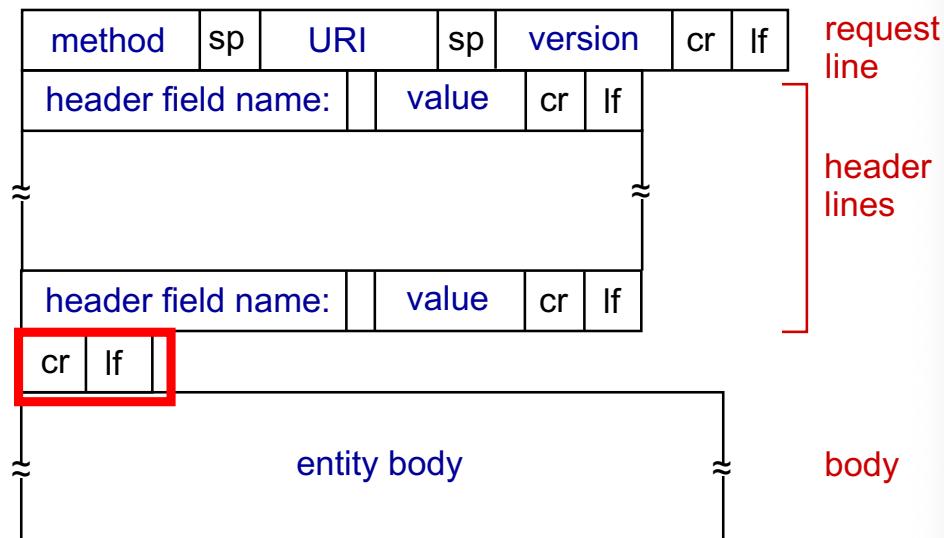
> Frame 20: 489 bytes on wire (3912 bits), 489 bytes captured (3912 bits) on interface en0
> Ethernet II, Src: Apple_74:26:5d (f4:d4:88:74:26:5d), Dst: Tp-LinkT_0b:d8:5c (00:0c:29:0b:d8:5c)
> Internet Protocol Version 4, Src: 192.168.0.24, Dst: 106.14.20.246
> Transmission Control Protocol, Src Port: 59244, Dst Port: 80, Seq: 1, Ack: 1, Len: 489
< Hypertext Transfer Protocol
  > GET / HTTP/1.1\r\n
  Host: www.feimax.com\r\n
  Connection: keep-alive\r\n
  Upgrade-Insecure-Requests: 1\r\n
  User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.122 Safari/537.36\r\n
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8\r\n
  Accept-Encoding: gzip, deflate\r\n
  Accept-Language: zh\r\n
  \r\n
  [Full request URI: http://www.feimax.com/]
  [HTTP request 1/2]
  [Response in frame: 22]
  [Next request in frame: 24]
```

Show packet bytes

Help



HTTP request message: general format



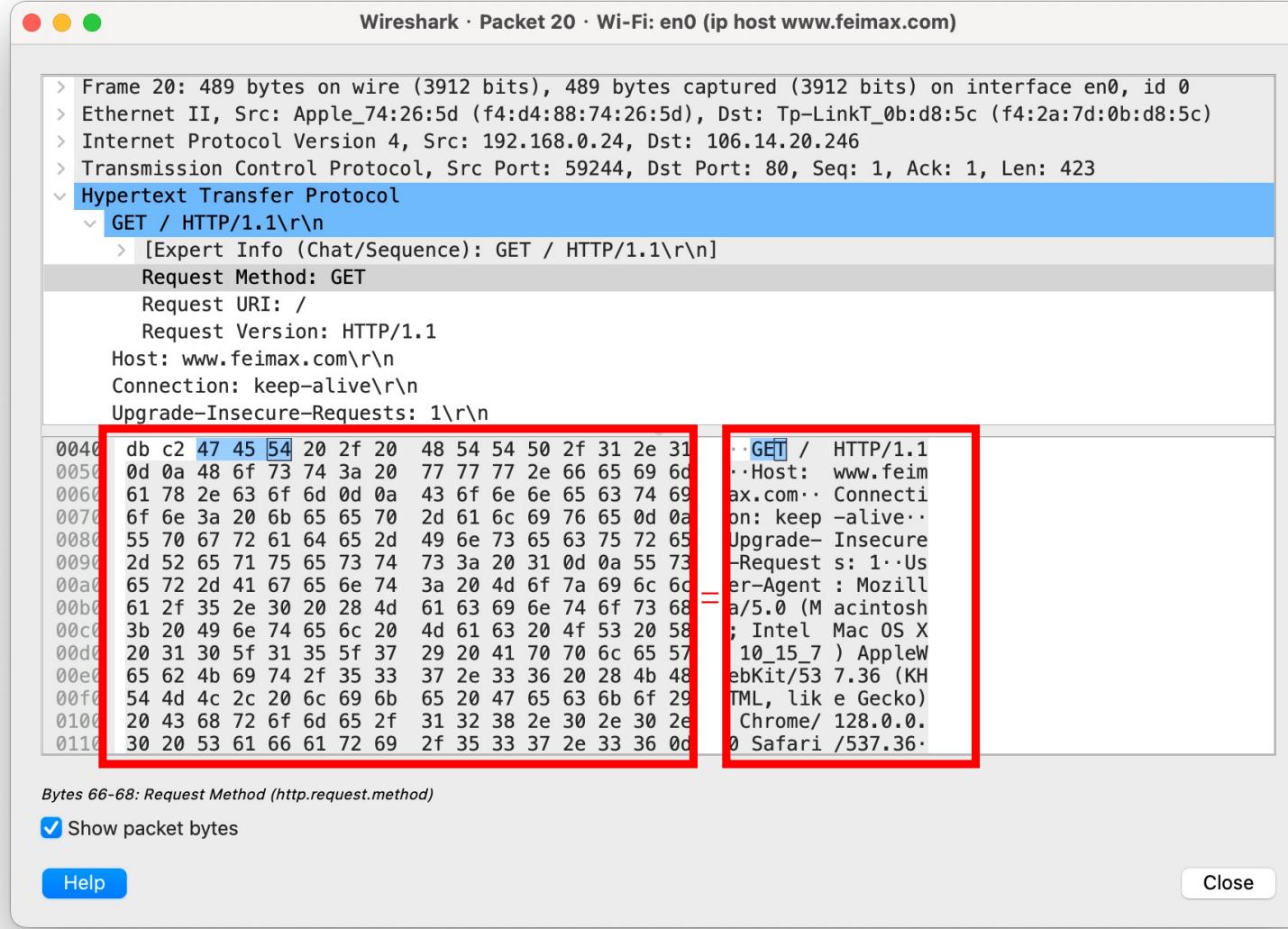
Wireshark · Packet 20 · Wi-Fi: en0 (ip host www.feimax.com)

```

> Frame 20: 489 bytes on wire (3912 bits), 489 bytes captured (3912 bits) on interface
> Ethernet II, Src: Apple_74:26:5d (f4:d4:88:74:26:5d), Dst: Tp-LinkT_0b:d8:5c (00:0c:29:0b:d8:5c)
> Internet Protocol Version 4, Src: 192.168.0.24, Dst: 106.14.20.246
> Transmission Control Protocol, Src Port: 59244, Dst Port: 80, Seq: 1, Ack: 1,
< Hypertext Transfer Protocol
  > GET / HTTP/1.1\r\n
    Host: www.feimax.com\r\n
    Connection: keep-alive\r\n
    Upgrade-Insecure-Requests: 1\r\n
    User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.122 Safari/537.36
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
    Accept-Encoding: gzip, deflate\r\n
    Accept-Language: zh\r\n
  \r\n
  [Full request URI: http://www.feimax.com/]
  [HTTP request 1/2]
  [Response in frame: 22]
  [Next request in frame: 24]

 Show packet bytes
Help

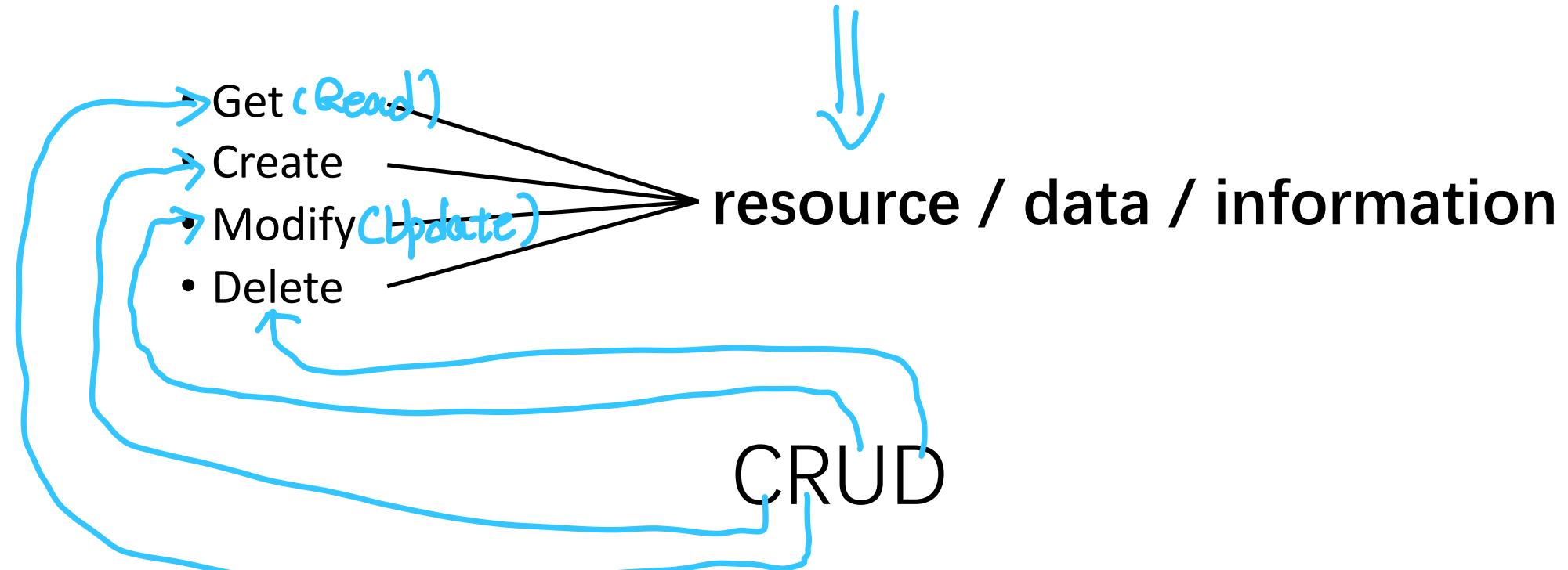
```



Human Readable!

HTTP 1.1 request method

- What do we need from a web protocol?



不同颜色对应

HTTP 1.0 1.1 Additional methods

- GET
- POST
 > 上传
- PUT
- DELETE
- PATCH
 更新.

对特定资源获取
选项
方法

- HEAD (检查资源是否存在)
- TRACE 测试请求在传输前是否被修改
- OPTIONS
- CONNECT 代理服务器SSL加密通信

HTTP 1.1 request **headers**

- Key-value pairs, metadata about the request and the client:

- Host name
- Authentication
- Content types
- User-agent information
- Caching / Cookies
- Types of connections
- ...

Types of connections

非持久

Non-persistent HTTP

- At most one object sent over TCP connection
 - connection then closed
- Downloading multiple objects required multiple connections

持久

Persistent HTTP

- Multiple objects can be sent over single TCP connection between client, server

Non-persistent HTTP: response time

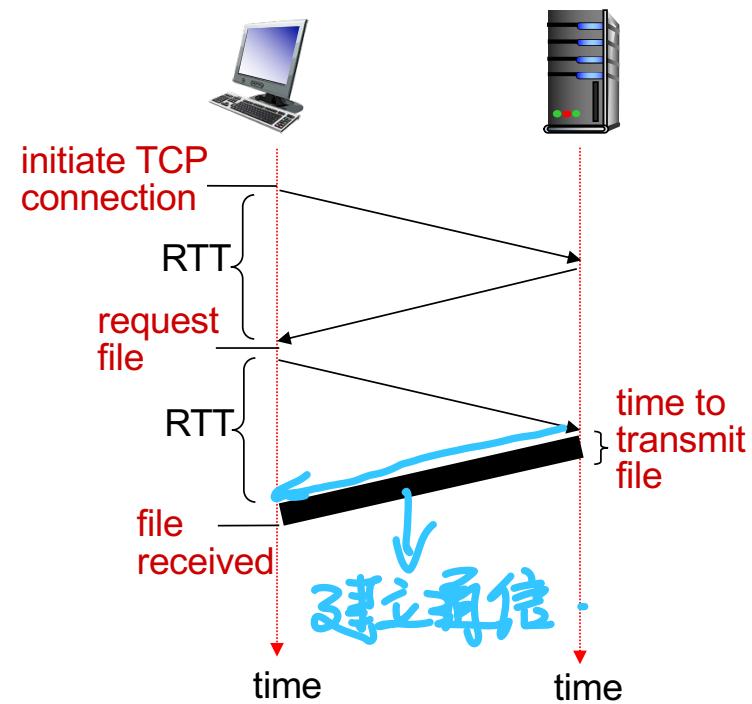
RTT (Round Trip Time): time for a small packet to travel from client to server and back round trip time

HTTP response time:

- One RTT to initiate TCP connection
- One RTT for HTTP request and first few bytes of HTTP response to return
- File transmission time
- Non-persistent HTTP response time =

$2\text{RTT} + \text{file transmission time}$

请求文件需要传输时间



Persistent HTTP

Non-persistent HTTP issues:

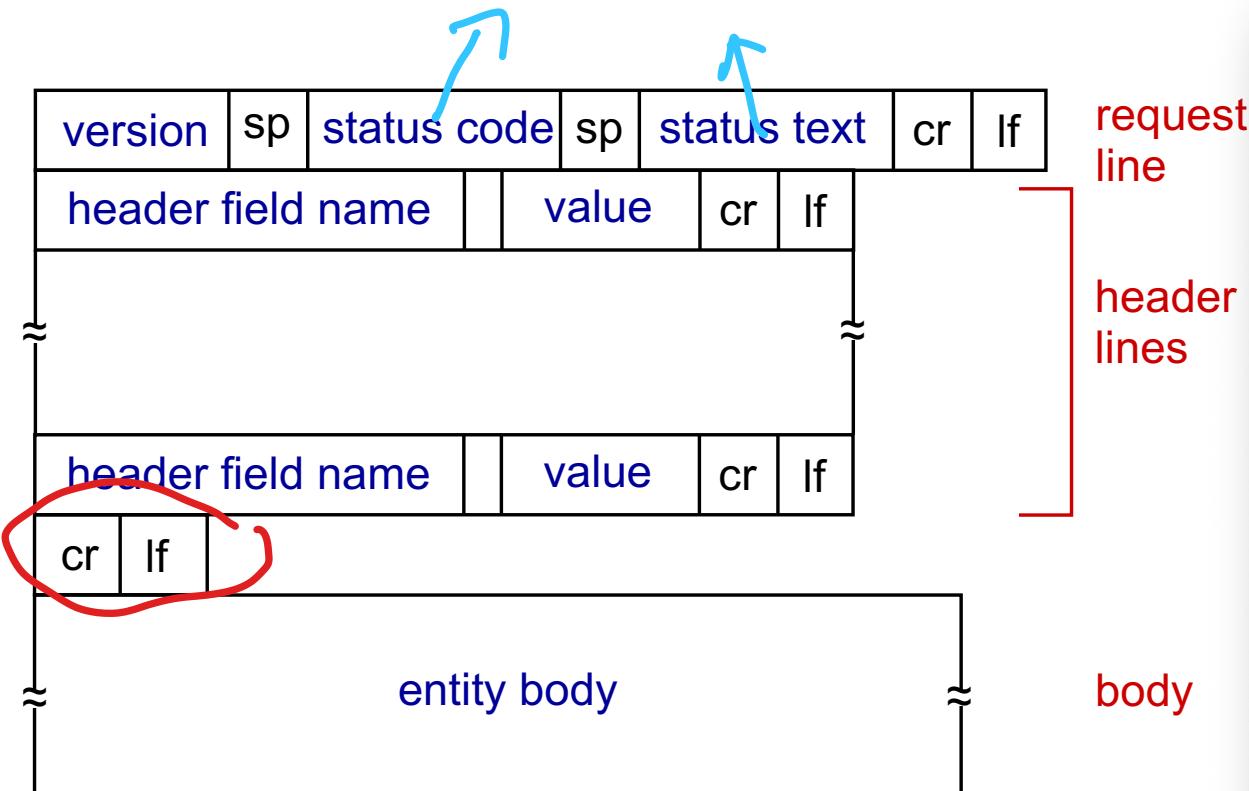
- Requires 2 RTTs per object
- OS overhead for each TCP connection
- Browsers often open parallel TCP connections to fetch referenced objects

Q: Is persistent HTTP perfect? <=

Persistent HTTP issues:

- Server leaves connection open after sending response
- Subsequent HTTP messages between same client/server sent over open connection
- Client sends requests as soon as it encounters a referenced object
- As little as one RTT for all the referenced objects

HTTP response message: format



Wireshark · Packet 22 · Wi-Fi: en0 (ip host www.feimax.com)

```

> Frame 22: 784 bytes on wire (6272 bits), 784 bytes captured (6272 bits)
> Ethernet II, Src: TP-Link_0b:d8:5c (f4:2a:7d:0b:d8:5c), Dst: Apple (08:00:27:00:00:00)
> Internet Protocol Version 4, Src: 106.14.20.246, Dst: 192.168.0.24
> Transmission Control Protocol, Src Port: 80, Dst Port: 59244, Seq: 1, Ack: 1, Len: 784
  Hypertext Transfer Protocol
    > HTTP/1.1 200 OK\r\n
      Server: nginx\r\n
      Date: Tue, 17 Sep 2024 12:51:15 GMT\r\n
      Content-Type: text/html\r\n
      Transfer-Encoding: chunked\r\n
      Connection: keep-alive\r\n
      Vary: Accept-Encoding\r\n
      Content-Encoding: gzip\r\n
      \r\n
      [HTTP response 1/2]
      [Time since request: 0.022098000 seconds]
      [Request in frame: 20]
      [Next request in frame: 24]
      [Next response in frame: 2888]
      [Request URI: http://www.feimax.com/]
    > HTTP chunked response
      Content-encoded entity body (gzip): 511 bytes -> 825 bytes
      File Data: 825 bytes
  Line-based text data: text/html (22 lines)
    <!DOCTYPE html>\r\n
  
```

Show packet bytes

Help

HTTP response status codes

- Status code appears in 1st line in server-to-client response message.
- Some sample codes:
 - 200 OK
 - 301 Move Permanently 永久移址 .
 - 400 Bad Request
 - 404 Not Found
 - 505 HTTP Version Not Supported

<https://www.w3.org/Protocols/rfc2616/rfc2616-sec6.html#sec6.1.1>

Status Code

- 1xx: Informational - Request received, continuing process
- 2xx: Success - The action was successfully received, understood, and accepted
- 3xx: Redirection - Further action must be taken in order to complete the request
- 4xx: Client Error - The request contains bad syntax or cannot be fulfilled
- 5xx: Server Error - The server failed to fulfill an apparently valid request

<https://www.w3.org/Protocols/rfc2616/rfc2616-sec6.html#sec6.1.1>

X

- 1xx: Informational - Request received, continuing process
- 2xx: Success - The action was successfully received, understood, and accepted
- 3xx: Redirection - Further action must be taken in order to complete the request
- 4xx: Client Error - The request contains bad syntax or cannot be fulfilled
- 5xx: Server Error - The server failed to fulfill an apparently valid request

Status-Code =
"100" ; Section [10.1.1](#): Continue
"101" ; Section [10.1.2](#): Switching Protocols
"200" ; Section [10.2.1](#): OK
"201" ; Section [10.2.2](#): Created
"202" ; Section [10.2.3](#): Accepted
"203" ; Section [10.2.4](#): Non-Authoritative Information
"204" ; Section [10.2.5](#): No Content
"205" ; Section [10.2.6](#): Reset Content
"206" ; Section [10.2.7](#): Partial Content
"300" ; Section [10.3.1](#): Multiple Choices
"301" ; Section [10.3.2](#): Moved Permanently
"302" ; Section [10.3.3](#): Found
"303" ; Section [10.3.4](#): See Other
"304" ; Section [10.3.5](#): Not Modified
"305" ; Section [10.3.6](#): Use Proxy
"307" ; Section [10.3.8](#): Temporary Redirect
"400" ; Section [10.4.1](#): Bad Request
"401" ; Section [10.4.2](#): Unauthorized
"402" ; Section [10.4.3](#): Payment Required
"403" ; Section [10.4.4](#): Forbidden
"404" ; Section [10.4.5](#): Not Found
"405" ; Section [10.4.6](#): Method Not Allowed
"406" ; Section [10.4.7](#): Not Acceptable
"407" ; Section [10.4.8](#): Proxy Authentication Required
"408" ; Section [10.4.9](#): Request Time-out
"409" ; Section [10.4.10](#): Conflict
"410" ; Section [10.4.11](#): Gone
"411" ; Section [10.4.12](#): Length Required
"412" ; Section [10.4.13](#): Precondition Failed
"413" ; Section [10.4.14](#): Request Entity Too Large
"414" ; Section [10.4.15](#): Request-URI Too Large
"415" ; Section [10.4.16](#): Unsupported Media Type
"416" ; Section [10.4.17](#): Requested range not satisfiable
"417" ; Section [10.4.18](#): Expectation Failed
"500" ; Section [10.5.1](#): Internal Server Error
"501" ; Section [10.5.2](#): Not Implemented
"502" ; Section [10.5.3](#): Bad Gateway
"503" ; Section [10.5.4](#): Service Unavailable
"504" ; Section [10.5.5](#): Gateway Time-out
"505" ; Section [10.5.6](#): HTTP Version not supported
extension-code

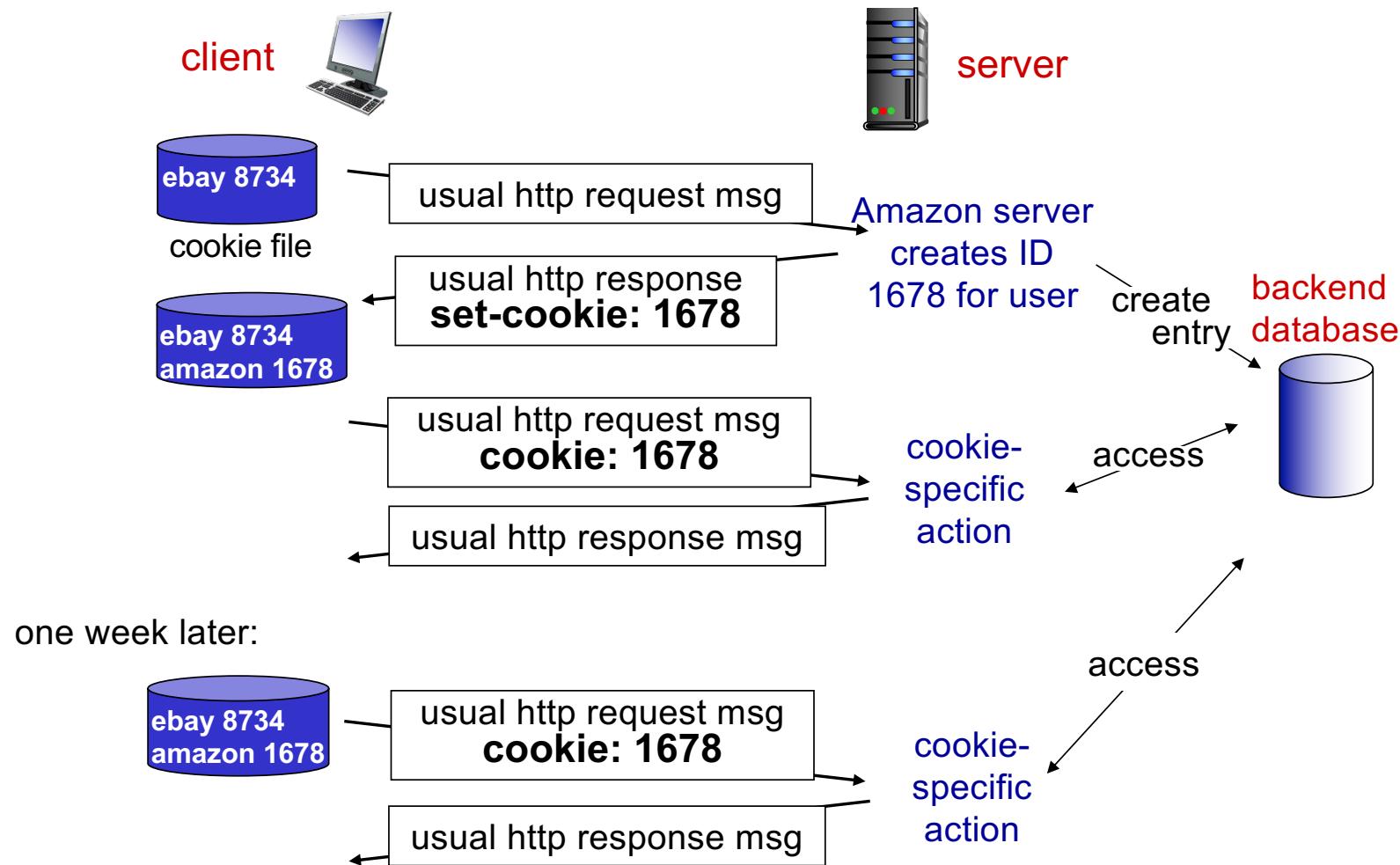
*HTTP is “stateless” !
So...*

User-server state: cookies

- Many Web sites use cookies
- Four components:
 1. cookie header line of HTTP **response** message
 2. cookie header line in next HTTP **request** message
 3. cookie file kept on user's host, managed by user's browser
 4. back-end database at Web site

Example:

- Bob always access Internet from PC
- Visits Taobao site for first time
- When initial HTTP requests arrives at site, site creates:
 - unique ID -> cookie
 - entry in backend database for ID



Cookies

What cookies can be used for:

- Authorization
- Recommendations
- User session state
- ...

aside

Cookies and privacy:

- Cookies permit sites to learn a lot about you
- Remember to clean your cookies

Wireshark Test

▼ Hypertext Transfer Protocol

► GET / HTTP/1.1\r\n

Host: www.feimax.com\r\n

Connection: keep-alive\r\n

Upgrade-Insecure-Requests: 1\r\n

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/77.0.3865.121 Safari/537.36

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3

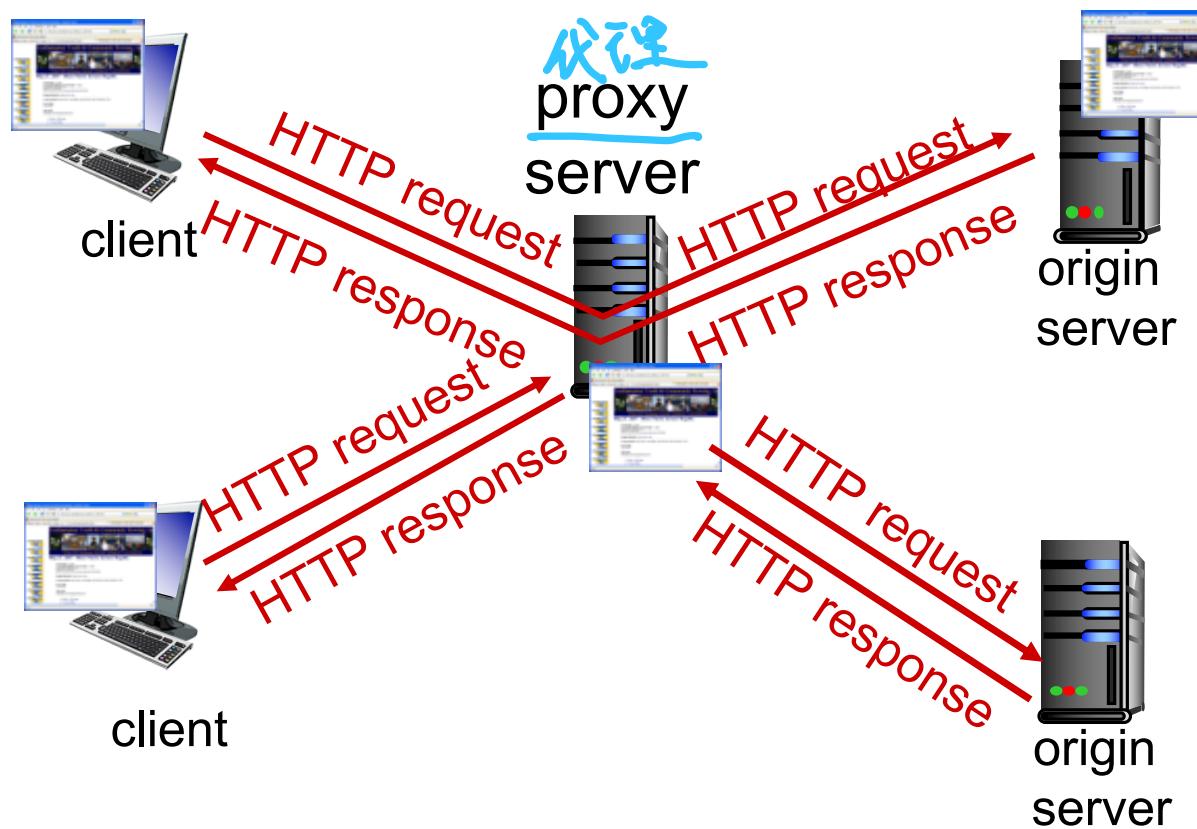
Purpose: prefetch\r\n

Accept-Encoding: gzip, deflate\r\n

Accept-Language: zh,en;q=0.9,zh-TW;q=0.8\r\n

► Cookie: wp-settings-time-1=1567750800; wp-settings-1=libraryContent%3Dbrowse%26in\r\n\r\n

Web caches



More about Web caching

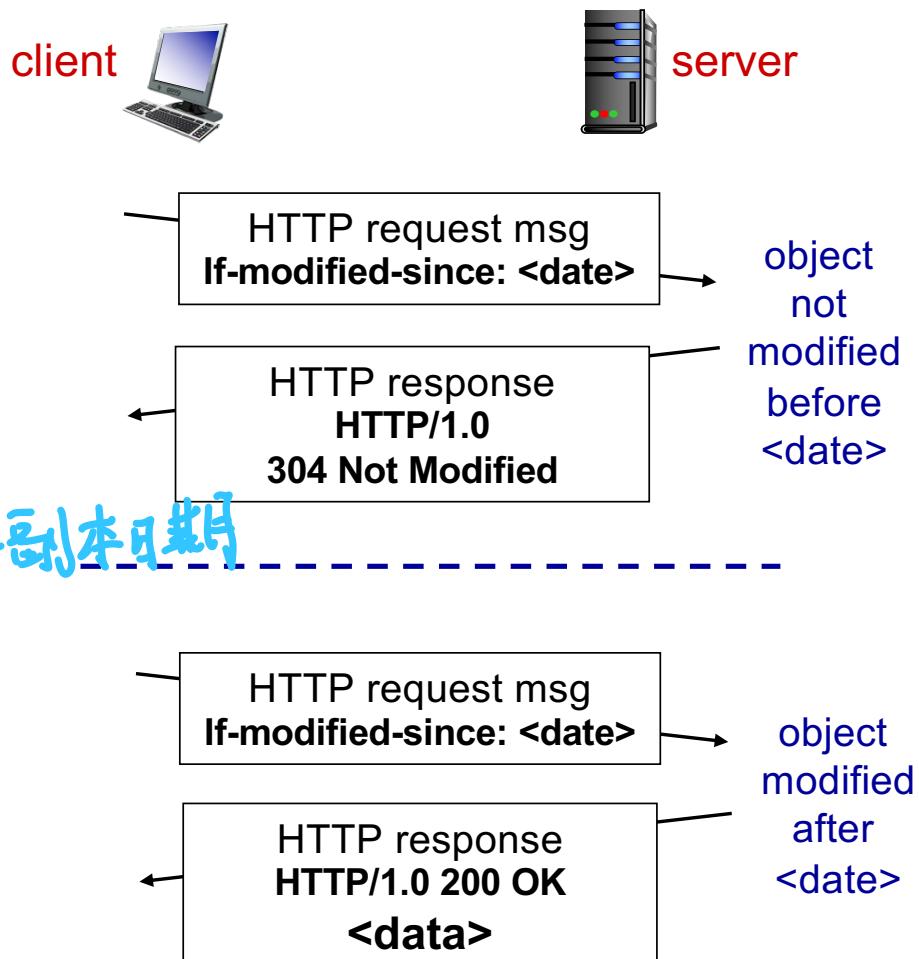
- Cache acts as both client and server
 - server for original requesting client
 - client to origin server
- Typically cache is installed by ISP (university, company, residential ISP)

Why Web caching?

- reduce response time for client request
- reduce traffic on an institution's access link
- Internet dense with caches: enables “poor” content providers to effectively deliver content (so too does P2P file sharing)

Conditional GET

- **Goal:** don't send object if cache has up-to-date cached version
 - no object transmission delay
 - lower link utilization
- **cache:** specify date of cached copy in HTTP request 指定 HTTP 请求中缓存副本日期
If-modified-since: <date>
- **server:** response contains no object if cached copy is up-to-date:
HTTP/1.0 304 Not Modified



HTTP = WEBPAGE?

Thanks.