

# 网络和套接字

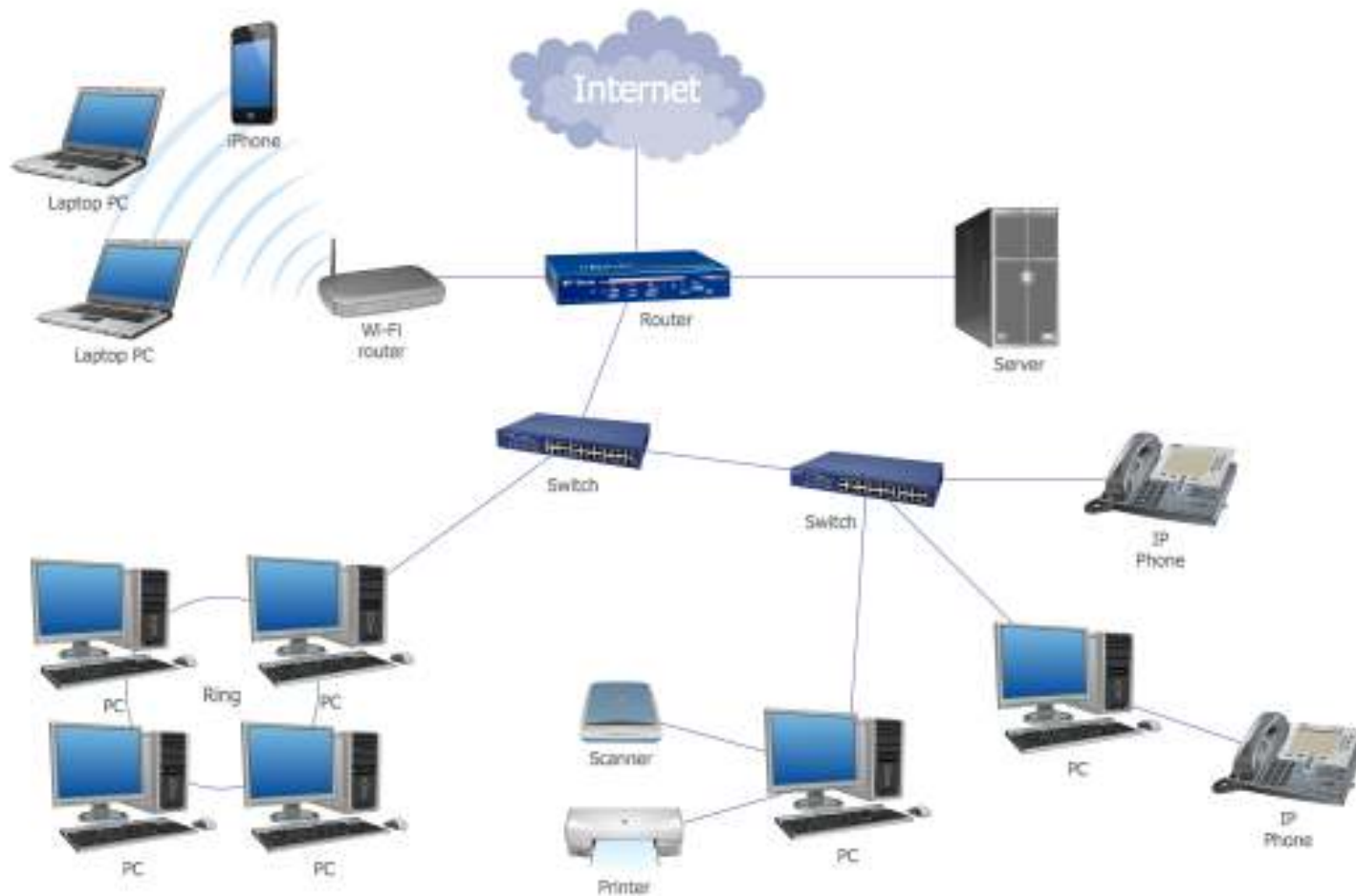
蒋炎岩

南京大学 | 计算机软件研究所 | 系统与软件分析研究组





# 网络：实现计算机之间的互联



# 问题：如何设计网络世界的API？

- 一个尝试
  - 给每个机器分配一个**机器号** (1, 2, 3, ...) – 类比pid
  - 每个机器可以建立和另一个机器之间的**管道**
    - `fd = network_pipe(machine_id);`
    - `read(fd, ...); write(fd, ...);`



# 一些分析



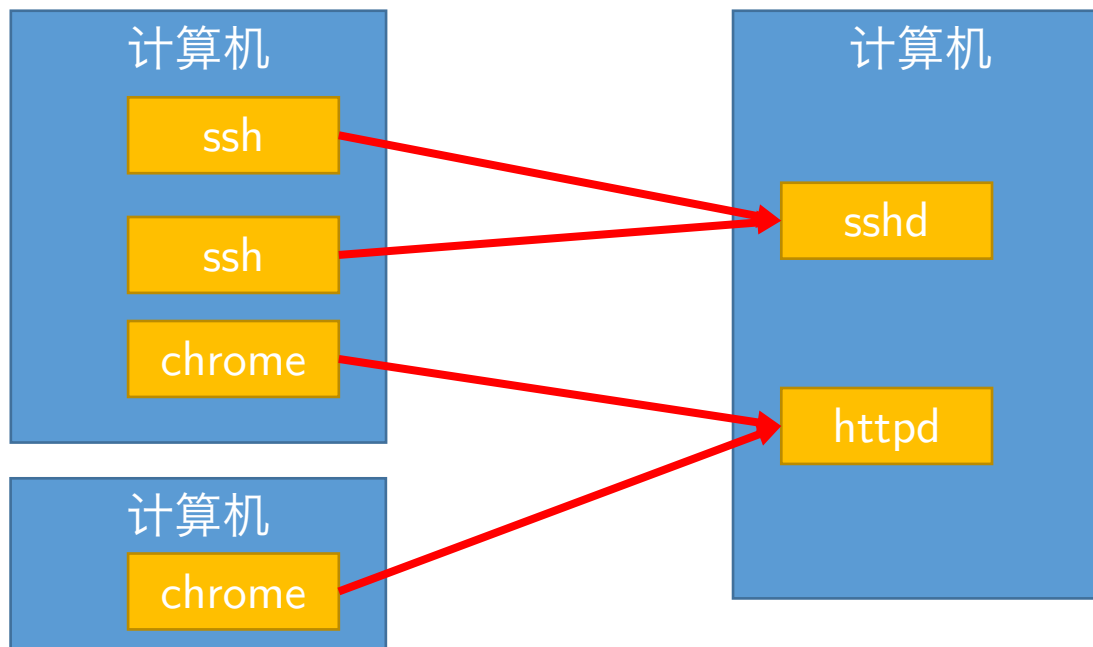
# 机器号模型：缺陷

- 在分布式系统上维护“机器号”
  - 用每个网卡唯一的id (MAC地址) → 升级了新网卡怎么办？
  - 自己起名字 → 重名怎么办？我冒充别人怎么办？
- 运行在一台计算机上的操作系统管理所有进程
  - 系统中每个进程都有唯一的pid



# 管道模型：缺陷1

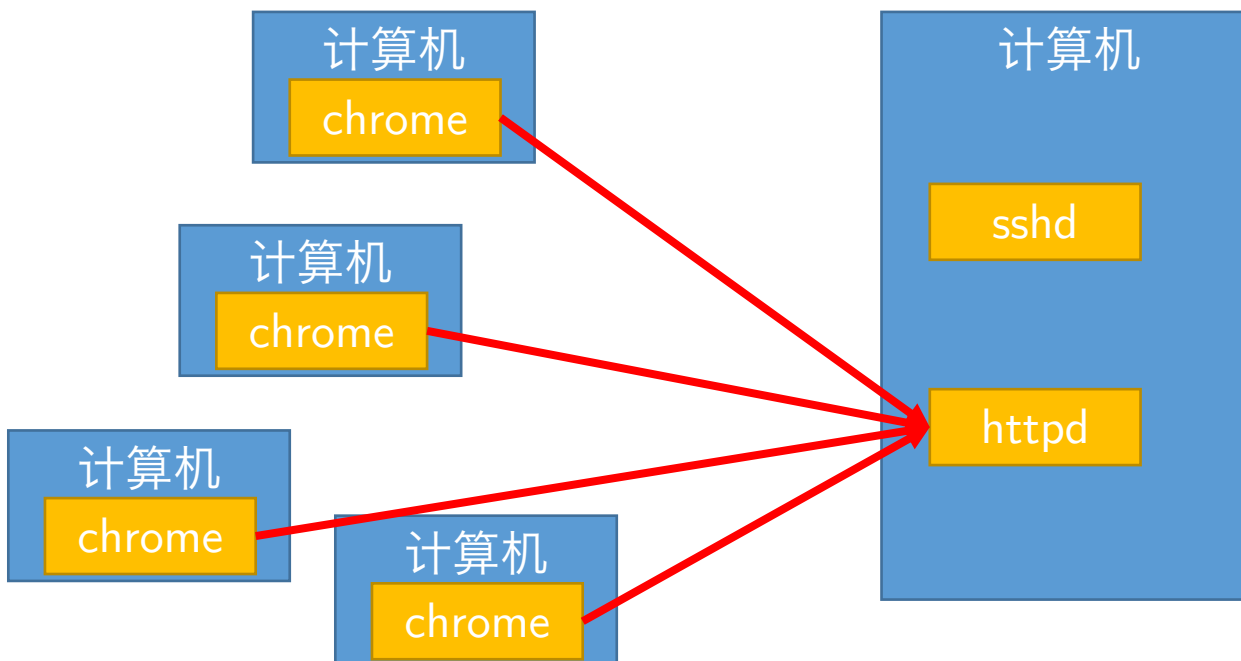
- 网络连接通常是应用到应用，而不是机器到机器的
  - 在管道上需要其他协议支持应用到应用的数据传输





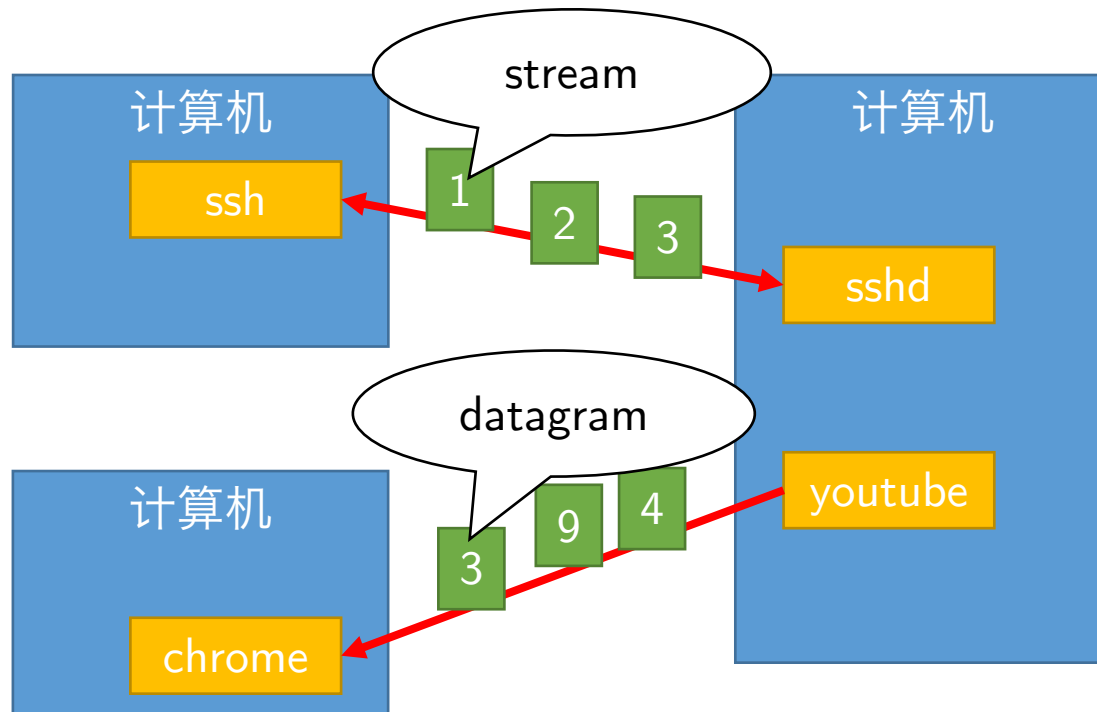
## 管道模型：缺陷2

- 应用可能需要管理大量的连接(天猫双十一)
  - 如何管理大量文件描述符？



## 管道模型：缺陷3

- 管道模型没有考虑到网络连接的一些特性
  - stream: 与管道类似，顺序的数据流
  - datagram: 一个一个“包”，按什么顺序到、到不到无所谓

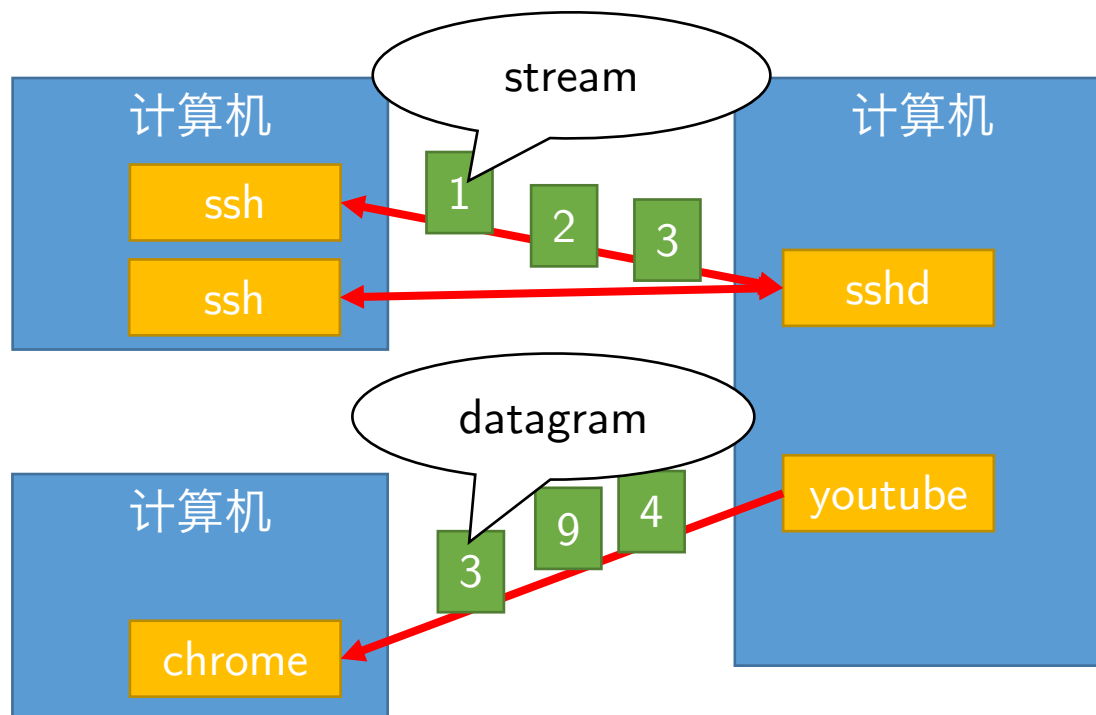




# 计算机网络简介

# 计算机网络：目标

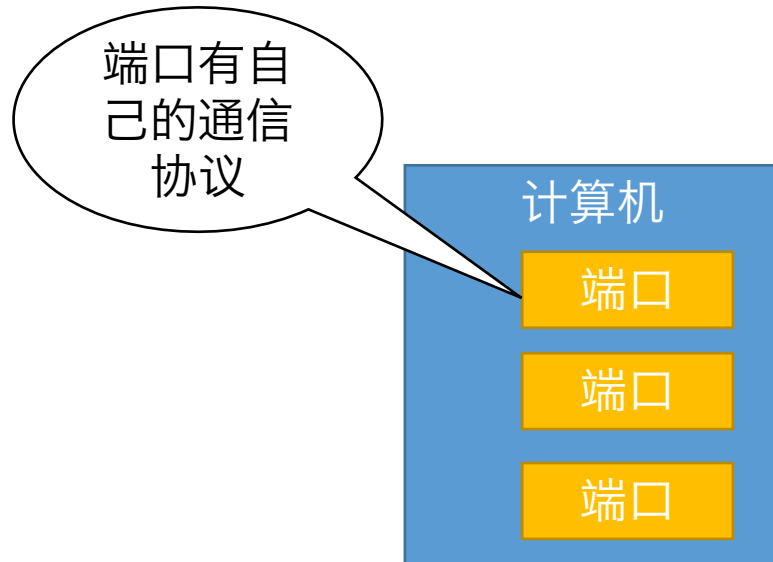
- 在应用和应用之间可靠地传递数据
  - 定位计算机上的应用
  - 实现数据的传递和控制





# 定位计算机：IP地址 (IP Address)

- 一些大家熟悉的例子
  - 课程网站：114.212.81.90
  - 百度：123.125.115.110  $\Leftrightarrow$  baidu.com
  - Github：github.com
- 名字 = 地址
  - 依靠域名解析(DNS)

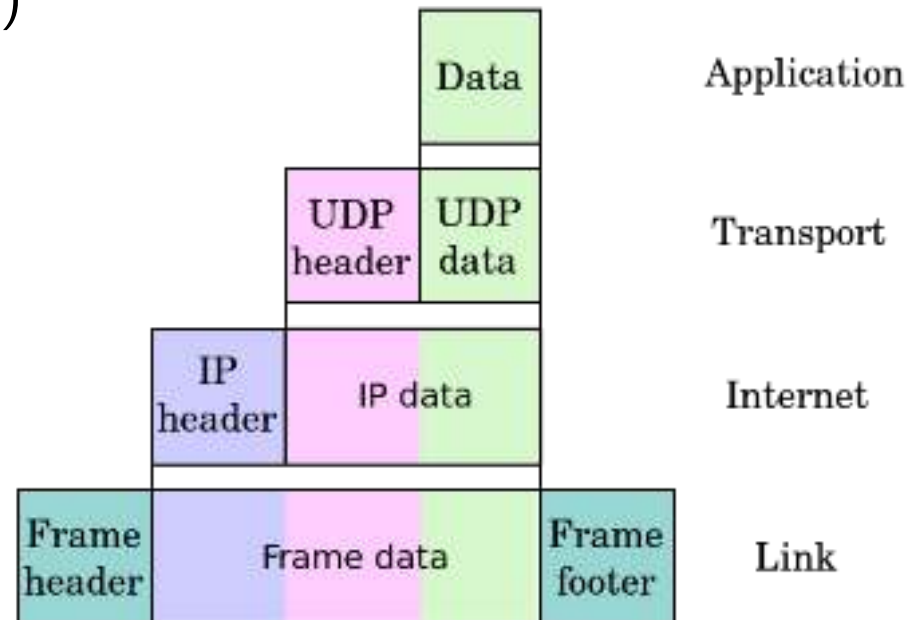


# 定位应用：端口 (Port)

- 一些大家熟悉的例子
  - 课程网站: 114.212.81.90:5000
  - 百度: 123.125.115.110:80  $\Leftrightarrow$  baidu.com:80
  - Github: github.com:443
- 应用 = 地址:端口 (1-65535)
  - IP地址: 127.0.0.1, 192.168.0.1, ...
  - 端口: 22, 80, 13459, ...
  - 完成了 “计算机上应用的定位”

# 传输数据 IPA:Port ⇔ IPA:Port

- TCP (Transmission Control Protocol)
  - 网络两端的可靠传输
- UDP (User Datagram Protocol)
  - 网络两端的best-effort传输
- 思考题：各有什么优缺点？
  - 考虑wikipedia/youtube应用



# 一些常见端口(网络应用)

	TCP	UDP	说明
7			Echo Protocol
20/21			FTP数据/命令传输
22			SSH (登录/端口转发)
53			DNS (域名服务)
80			HTTP (超文本)
161			SNMP (网络管理)
179			BGP (路由)
443			HTTPS (安全超文本)
3306			MySQL (数据库服务)
5801			VNC (远程桌面)
6379			Redis (数据库服务)

绿色：已被标准化；黄色：事实标准。

## 例子: netstat

- 查看课程网站的连接情况 (netstat -at)

```
Active Internet connections (servers and established)
Local Address           Foreign Address         State
*:ssh                   *::*                     LISTEN
*:5000                   *::*                     LISTEN
114.212.81.90:ssh       114.212.87.17:6828     ESTABLISHED
114.212.81.90:5000     114.212.87.17:5497     TIME_WAIT
```

ssh命令

课程网站(http)

浏览器

## 例子: nmap

- Nmap uses raw IP packets in novel ways to determine what hosts are available on the network, what services (application name and version) those hosts are offering, what operating systems (and OS versions) they are running, what type of packet filters/firewalls are in use, and dozens of other characteristics

```
# nmap -0 114.212.81.90
```

```
Starting Nmap 7.01 ( https://nmap.org )
```

```
Host is up (0.0017s latency).
```

```
Not shown: 961 closed ports, 37 filtered ports
```

```
PORT      STATE SERVICE
```

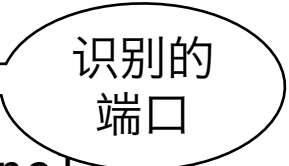
```
22/tcp    open  ssh
```

```
5000/tcp  open  upnp
```

```
Running: Linux 3.X|4.X
```

```
OS CPE: cpe:/o:linux:linux_kernel:3
```

```
cpe:/o:linux:linux_kernel:4
```



识别的  
端口

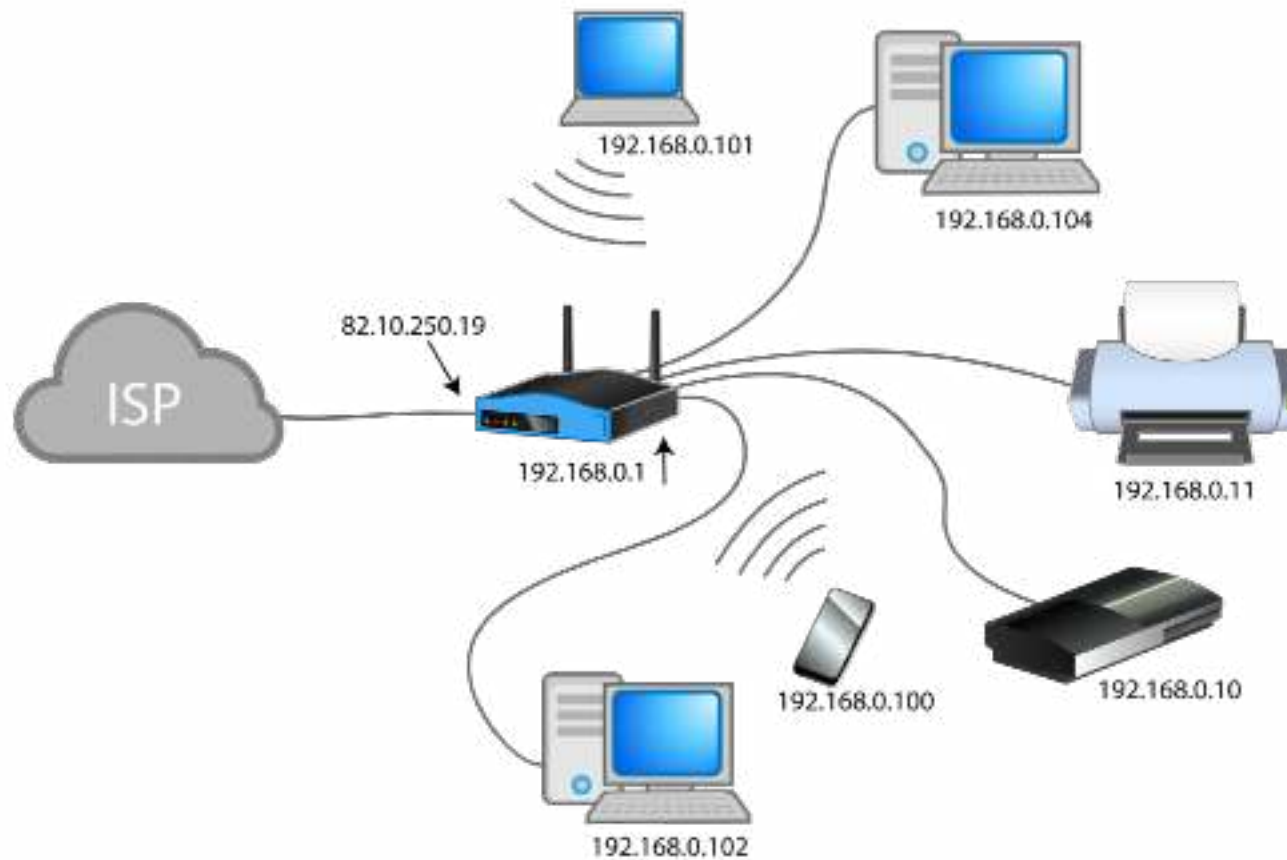


# 另一个问题：IP地址不够用啊！

- 至多只有 $255^4$ 个地址
  - 根本不够现在每台计算机用的，但计算机都**联网**了
  - 这是怎么解决的(除了使用更多的地址)?
- 有三类非常常见的IP地址(私有地址)是有重名的
  - 10.0.0.0 - 10.255.255.255 ← 在有些场景会看到
  - 172.16.0.0 - 172.31.255.255 ← 仙林的无线网
  - 192.168.0.0 - 192.168.255.255 ← 宿舍的路由器

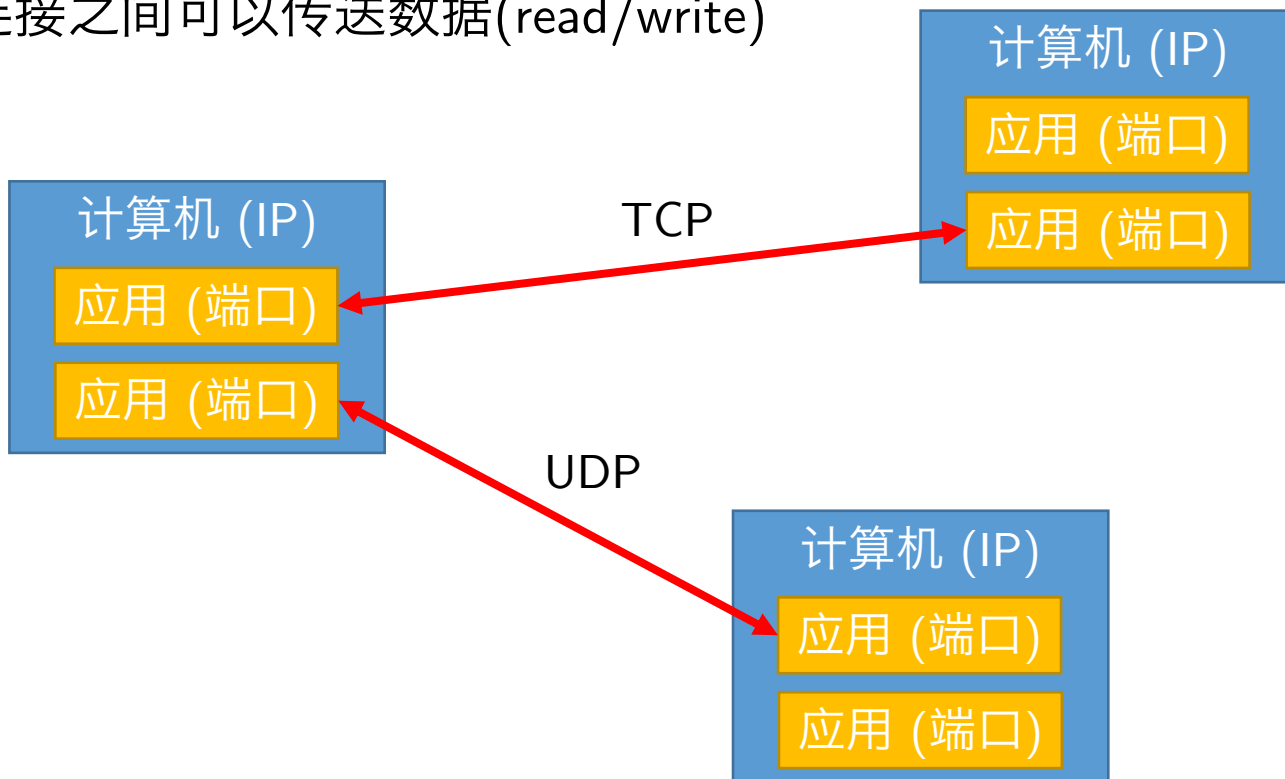


# NAT: 解决网络地址不够用



# 计算机网络：小结

- 计算机网络(超级简化模型)
  - 应用程序用IPA:Port表示
  - 一对IPA:Port之间能够建立连接(TCP/UDP)
  - 连接之间可以传送数据(read/write)



# IP Address + Port = Everything!



# 计算机网络：应用

# (1) Web服务器/浏览器

- Web服务器运行HTTP (Hypertext Transfer Protocol)
  - 默认是80 (http)/443 (https)端口
  - 但也可以运行在任何端口(114.212.81.90:5000)
- 传递的完全是文本信息
  - 若干行 “key: value” 形式，空行之后是实际内容
  - 图片、javascript、css都是这样传输的

# (1) Web服务器/浏览器 (cont'd)

浏览器发送

GET / HTTP/1.1

Host: 114.212.81.90:5000

User-Agent: curl/7.54.0

Accept: \*/\*

HTTP/1.0 302 FOUND

服务器发送

Content-Type: text/html; charset=utf-8

Content-Length: 237

Location: http://114.212.81.90:5000/wiki/Main\_Page

Server: Werkzeug/0.14.1 Python/3.5.2

Date: Sun, 03 Jun 2018 11:50:59 GMT

<title>Redirecting...</title>

<h1>Redirecting...</h1>

<p>You should be redirected automatically to target

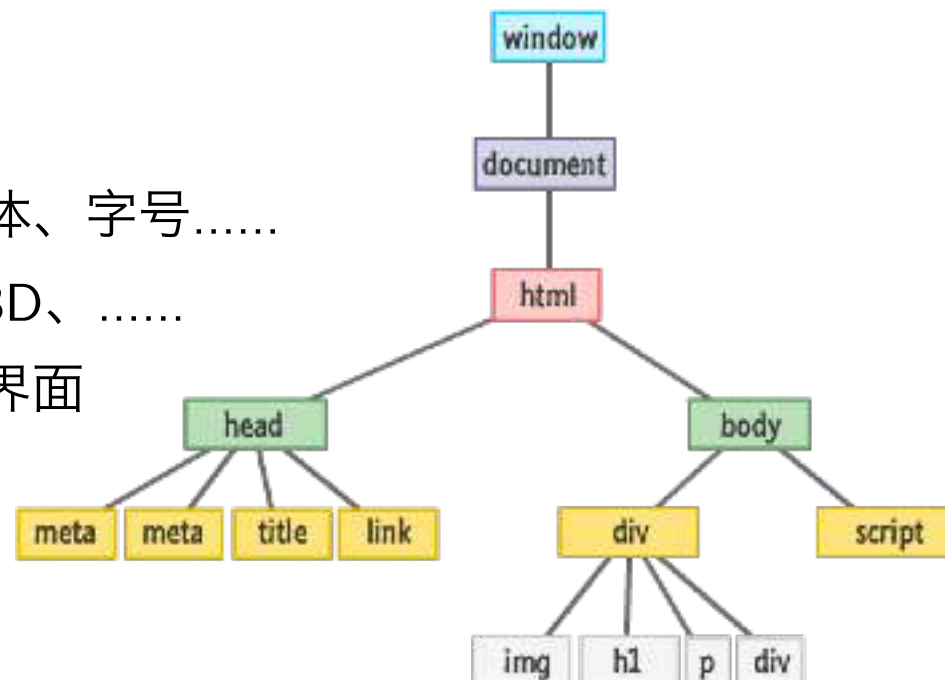
URL: <a href="/wiki/Main\_Page">[/wiki/Main\\_Page](/wiki/Main_Page)</a>.



# 什么？这就是我们每天沉迷的浏览器？

- DOM Tree

- 定义元素的大小、位置、字体、字号.....
- 可以有图片、声音、动画、3D、.....
- 可以用于很容易地展示任何界面



- Scripts

- 网页中包含脚本
- 脚本能发送/接收(AJAX)更多的请求、更新DOM Tree
- JavaScript已经是目前最受欢迎的语言了
  - 得益于Google在浏览器技术的长足进步和开源社区的活跃

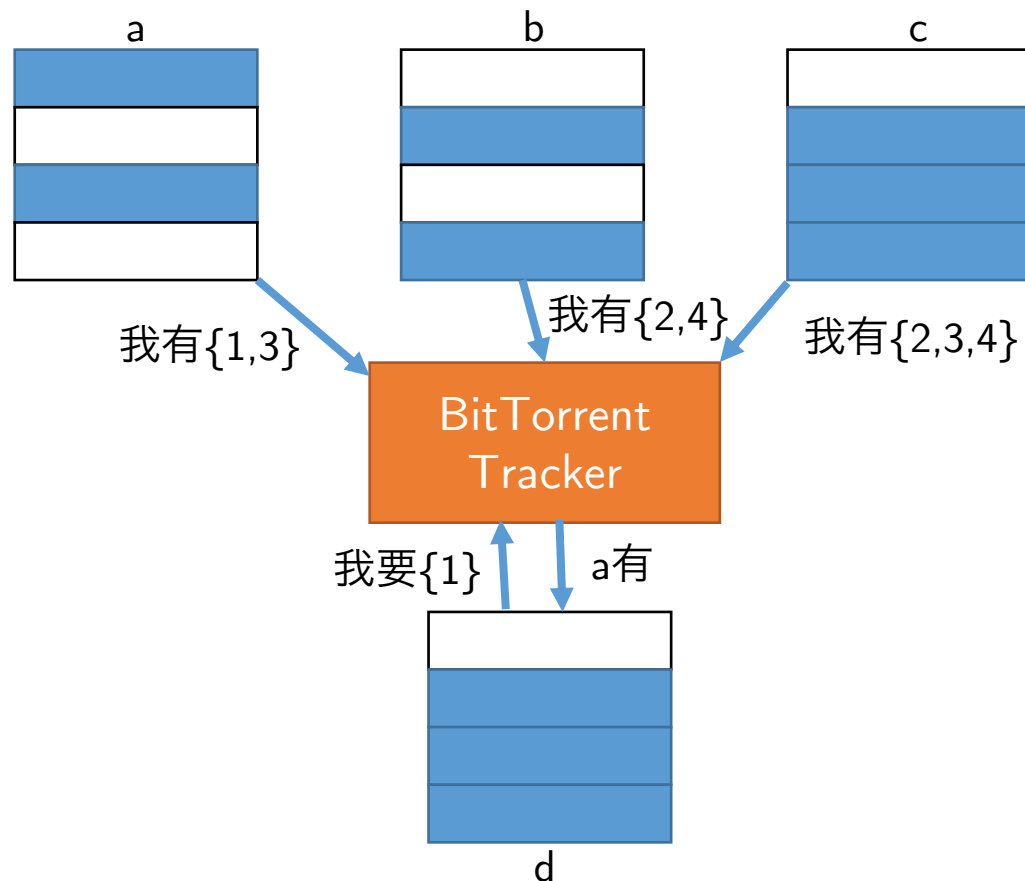


## (2) BitTorrent

- Client/Server通常是“下载的人越多越慢”
  - 服务器的带宽、处理速度有瓶颈
  - 这世上还有“**下载的人越多越快**”的好事？
- 人人都既当Client又当Server，互相交换数据就好啦

## (2) BitTorrent (cont'd)

- Tracker: 维护元数据
  - 每个节点都可以作为tracker





# The Open Napster (OpenNap) Protocol

- 与HTTP类似
  - 规定服务器运行在8888/7777端口
  - 定义了消息的格式和语义(允许Client-Client直接传输)
- 只要所有人遵守这个协议就行
  - 服务器为了维持公平, 也会做一些限制
    - 例如紫荆的上传量统计(如何公平的统计?)

Specification: <http://opennap.sourceforge.net/napster.txt>

## (3) Block Chain

- 由全世界人维护的一个 `std::vector<const std::string>`
  - 仅支持 `push_back` 和 `const operator []` (不可篡改)
  - 每个 `string` 都是一条消费记录，就构成了 Bitcoin
- 但这个数据结构有并发的 `push_back`
  - 公平随机地选出一个人做 “主维护者”
  - 所有人和这个人同步



网络通信：API

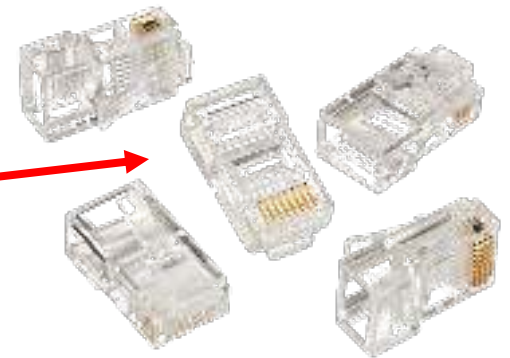


# 网络API解决的问题

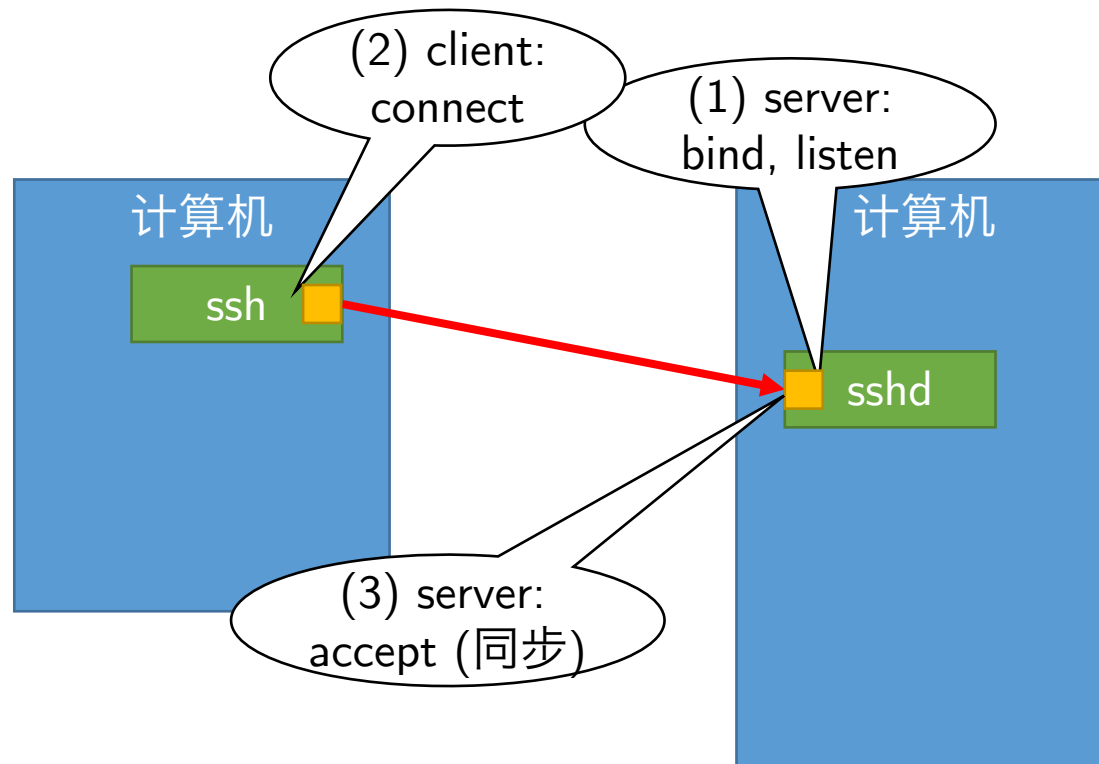
- 设计一组API，实现
  - 应用到应用的**大量**网络连接管理
  - **兼容各种数据传输模式**
- 这组API还**顺便**解决进程间通信的问题
  - 匿名管道只能用于父子进程
  - 命名管道需要另外的机制管理

# Berkeley Sockets (POSIX Sockets, WinSock)

- “A socket is an abstract representation (handle) for the local endpoint of a network communication path”
- socket (套接字)也是操作系统中的一个对象
  - 通过套接字访问网络连接的一端
  - 可以随意创建
    - Domain: AF\_INET, AF\_INET6, AF\_UNIX
    - Type: SOCK\_STREAM, SOCK\_DGRAM, SOCK\_SEQPACKET, SOCK\_RAW
  - 用API实现套接字之间的连接
    - 可以在一个计算机内，也可以跨越计算机



# TCP/IP Sockets







# 一个最简HTTP服务器

```
import sys, socket
```

创建

```
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

```
sock.bind(('0.0.0.0', 8000))
```

绑定

```
sock.listen(1)
```

监听

```
while True:
```

```
    conn, _ = sock.accept()
```

连接

```
    data = conn.recv(1024)
```

```
    reply = '<h1>It works!</h1>'
```

```
    response = '\n'.join([
```

```
        'HTTP/1.0 200 OK',
```

```
        'Content-Type: text/html; charset=utf-8',
```

```
        'Content-Length: {}'.format(len(reply)),
```

```
        '',
```

```
        reply])
```

```
    conn.sendall(response)
```

```
    conn.close()
```

# 这(几乎)就是网络世界的全部

- 使用套接字建立应用之间的连接、传输数据
  - 在传输数据基础上构建各种应用协议

