点击一个链接后发生了什么?

- 浏览器输入 https://www.kaola.com,这是一个 URL，通过链接不能知道要访问的地址，所以它不知道如何访问，于是它打开地址簿去查找。可以使用 **DNS** 去查找，也可以使用另一种更加精准的查找协议 HTTPDNS。
- 最终得到了这个地址 106.114.138.24。这是一个 IP 地址，是互联网世界的“门牌号”。
- 知道了目标地址，浏览器就开始打包它的请求，普通浏览请求通常会使用 HTTP 协议；但对于购物请求，往往需要加密传输，因而使用 HTTPS 协议，无论什么协议，内容里都要写上“你要买什么和买多少”

DNS,HTTP,HTTPS 所在的层我们称为**应用层。**经过应用层封装后，浏览器会将应用层包交给下一层（传输层）处理，通过 socket 编程来实现。

传输层有两种协议，一种是无连接的协议 UDP，一种是面向链接的协议 TCP。对于支付来讲，通常使用 TCP 协议。所谓的面向连接就是，TCP 会保证这个包能够到达目的地，如果不能到达，就会重新发送，直到到达。

TCP 协议里会有两个端口，一个是浏览器监听的端口，一个是电商服务器监听的端口。操作系统通过端口来判断，它得到的包应该给那个进程。

传输层封装完毕后，浏览器会将包交给操作系统的**网络层**，网络层的协议是 IP 协议。

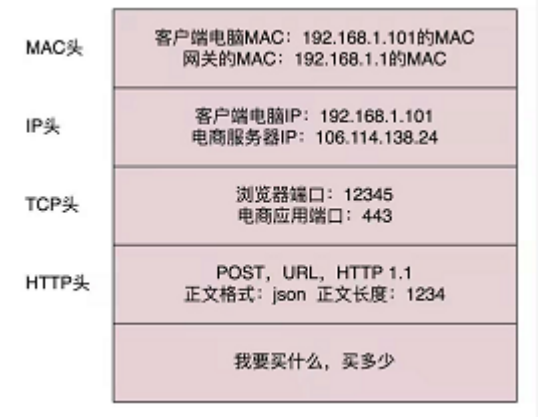
在 IP 协议里会有源 IP 地址，即浏览器所在机器的 IP 地址和目标 IP 地址，即电商网站所在服务器的 IP 地址。

操作系统直到了目标 IP 地址，开始想如何根据这个门牌号找到目标机器。操作系统往往会判断，这个目标 IP 地址是本地人还是外地人。如果是本地人，从门牌号就能看出来，显然电商网站在外地。

操作系统知道要离开本地去远方，虽然不知道远方在何处，但是可以类比人类，如果要去国外要去海关，去外地就要去**网关**。

操作系统启动的时候，就会被 DHCP 协议配置 IP 地址，以及默认的网关的 IP 地址 192.168.1.1

操作系统如何将 IP 地址发送给网关，在本地通信基本靠吼，于是操作系统大吼，谁是 192.168.1.1，网关会回答它，我就是，我的本地地址在本村东头。这个本地地址就是 **MAC** 地址，而大吼的那声就是 **ARP** 协议。

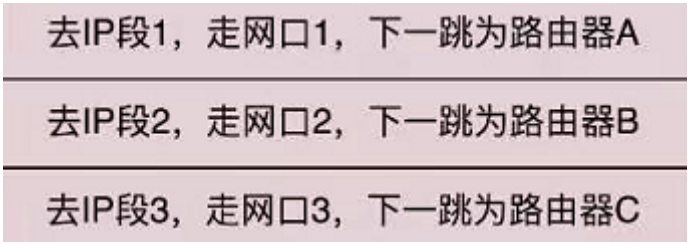


操作系统将 IP 包给了下一层，也就是 MAC 层。网卡再将包发出去，由于这个包里面是有 MAC 地址的，因而它能过够到达网关。

网关收到包之后，会根据自己的知识，判断下一步怎么走。网关往往是一个路由器，到某个 IP 地址怎么走，这个叫路由器表。

路由器有点像玄奘西行路过的一个个国家的一个个城关。每个城关都连接着两个国家，每个国家相当于一个局域网，在每个国家内部，都可以使用本地的 MAC 地址进行通信。

经过城关的时候，需要拿出 IP 地址，里面写着源 IP 地址和目标 IP 地址，然后询问加下来该怎么走？



城关往往是知道这些“知识”的，因为城关和临近的城关会经常沟通，到哪里应该怎么走，这种沟通的协议称为**路由协议**，常用的有 OSPF 和 BGP。

当网络包知道了下一步去那个城关，还是要使用国家内部的 MAC 地址，通过下一个城关的 MAC 地址，找到下一个城关，然后再问下一步路怎么走，一直到走出最后一个城关。

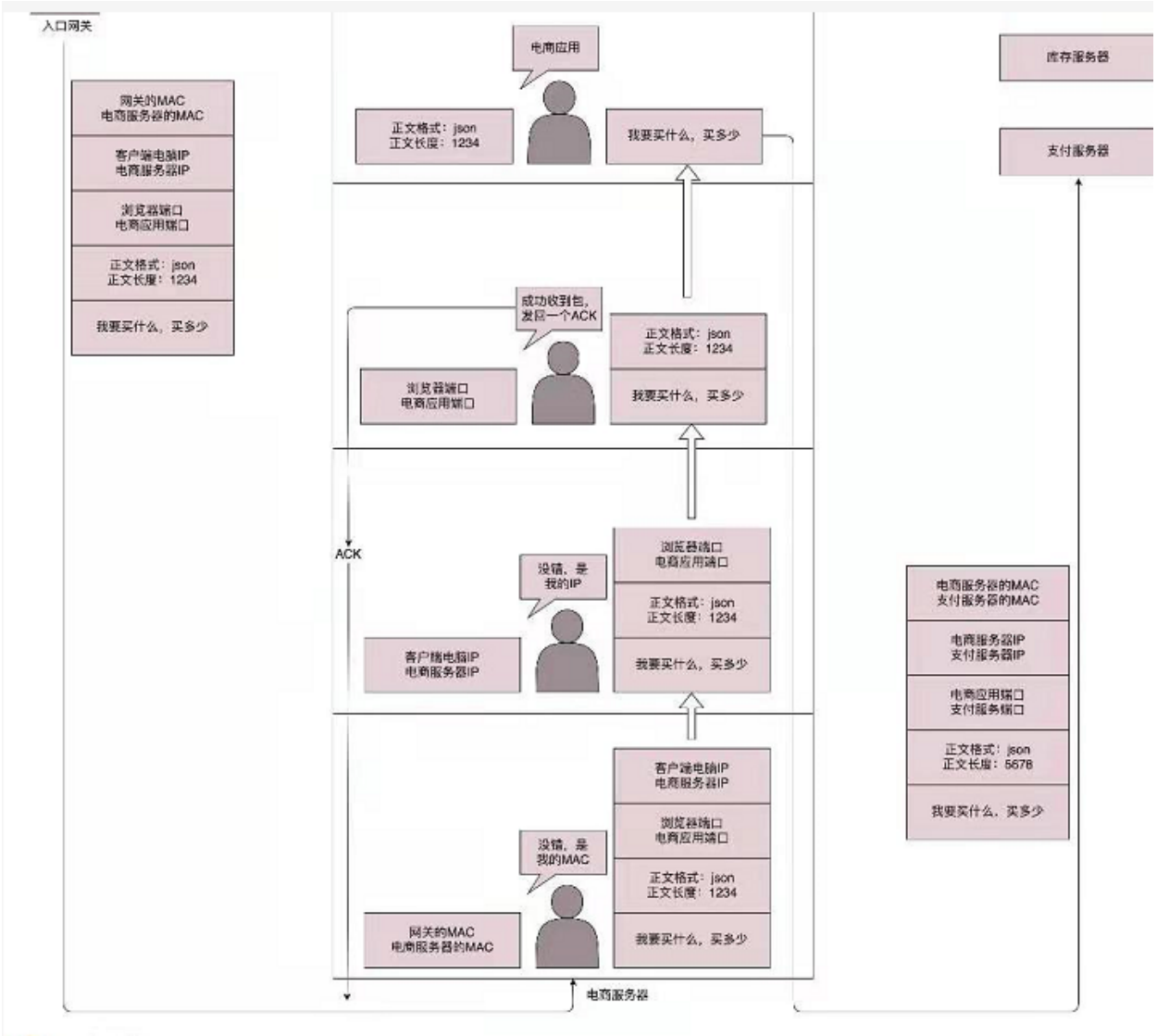
最后一个城关知道这个网络包要取的地方，于是，对着这个国家吼一声，这是谁的 IP 啊，目标服务器就会回复一个 MAC 地址。网络包过关后，就能通过这个 MAC 找到目标服务器。

目标服务器发现 MAC 地址对上了，取下 MAC 头来，发送给操作系统的网络层。发现 IP 地址也对上了，就取下 IP 头。IP 头里会写上一层封装的是 TCP 协议，然后将其交给传输层，即 TCP 层。

在这一层，对于每个收到的包，都会有一个回复的说明收到了。这个回复的包不是这次下单请求的结果，例如购物是否成功，扣了多少钱，而仅仅是 TCP 层的一个回复说明，这个说明会沿着刚才来的方向回去报信。（因为发送的过程中，包可能会丢失）

如果过一段时间还是没有收到，发送端的 TCP 层会重新发送这个包，还是上面的过程，直到收到平安到达的回复。**这个重试绝非你的浏览器重新将下单这个动作重新请求一遍。** 对于浏览器来讲，就发送了一次请求， TCP 层不断自己去重试。除非 TCP 层出了问题，例如连接断了，才轮到浏览器的应用层重新发送下单请求。

当网络包平安到达 TCP 层后， TCP 头中有目标端口号，通过端口号，可以找到电商网站进程正在监听这个端口号，假设一个 Tomcat，将这个包转发电商网站。



电商网站的进程得到 HTTP 请求的内容，知道了要买东西，买多少。往往一个电商网站最初的接待请求的这个 Tomcat 只是一个接待员，例如：负责告诉专门管理订单的进程，登记要买某个商品，买多少，要告诉管理库存的进程，库存要减少多少，要告诉支付的进程，应该付多少钱。

如何告诉相关的进程呢？通过 RPC 调用。远程过程调用就是当告诉管理订单进程的时候，接待员不用关心中间的网络互联问题，会由 RPC 框架统一处理。

RPC 框架有很多种，有基于 HTTP 协议放在 HTTP 的报文中的，有直接封装在 TCP 报文里的。

当接待员发现相应的部门处理完毕，就回复一个 HTTPS 的包，告知下单成功。这个 HTTPS 的包，会和来的时候一样到达个人电脑，最终进入浏览器，显示支付成功。