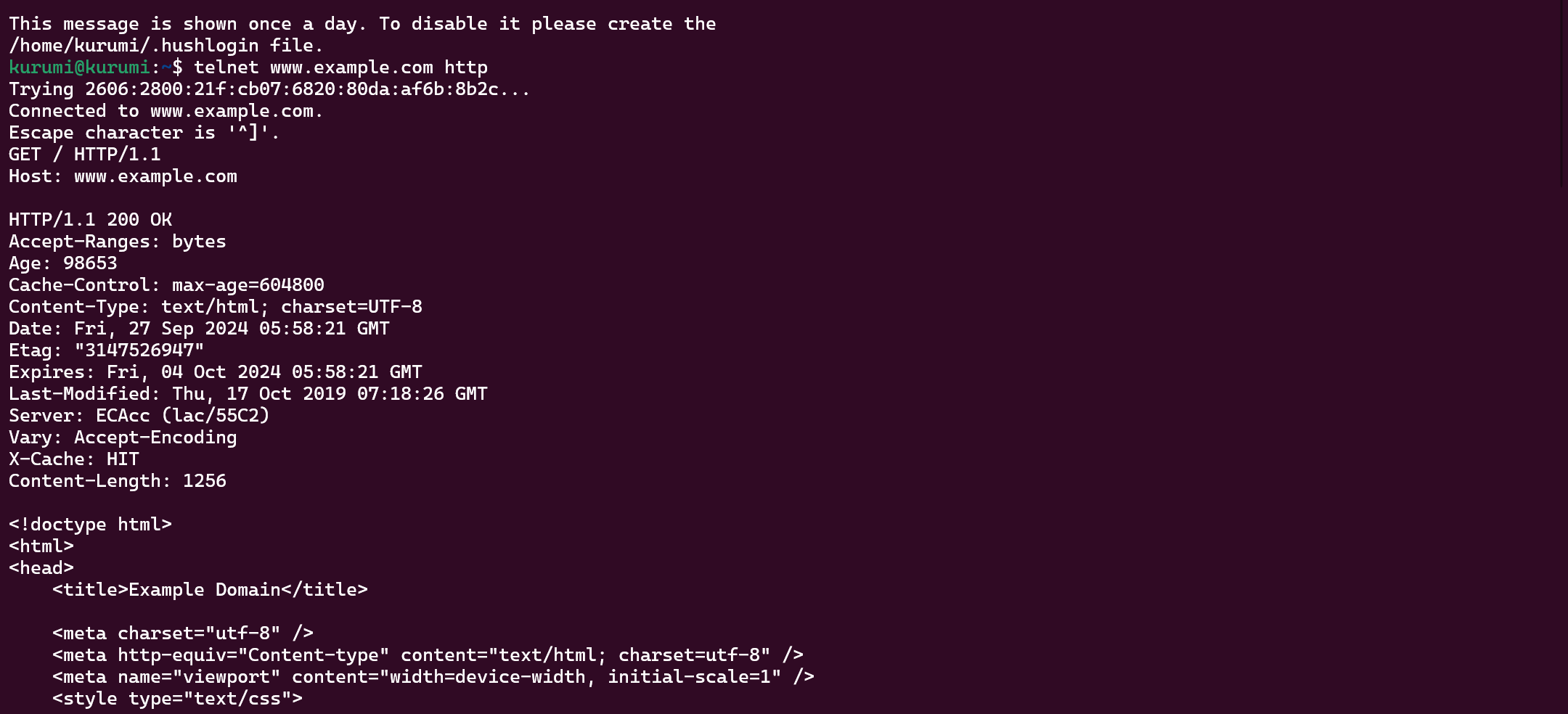
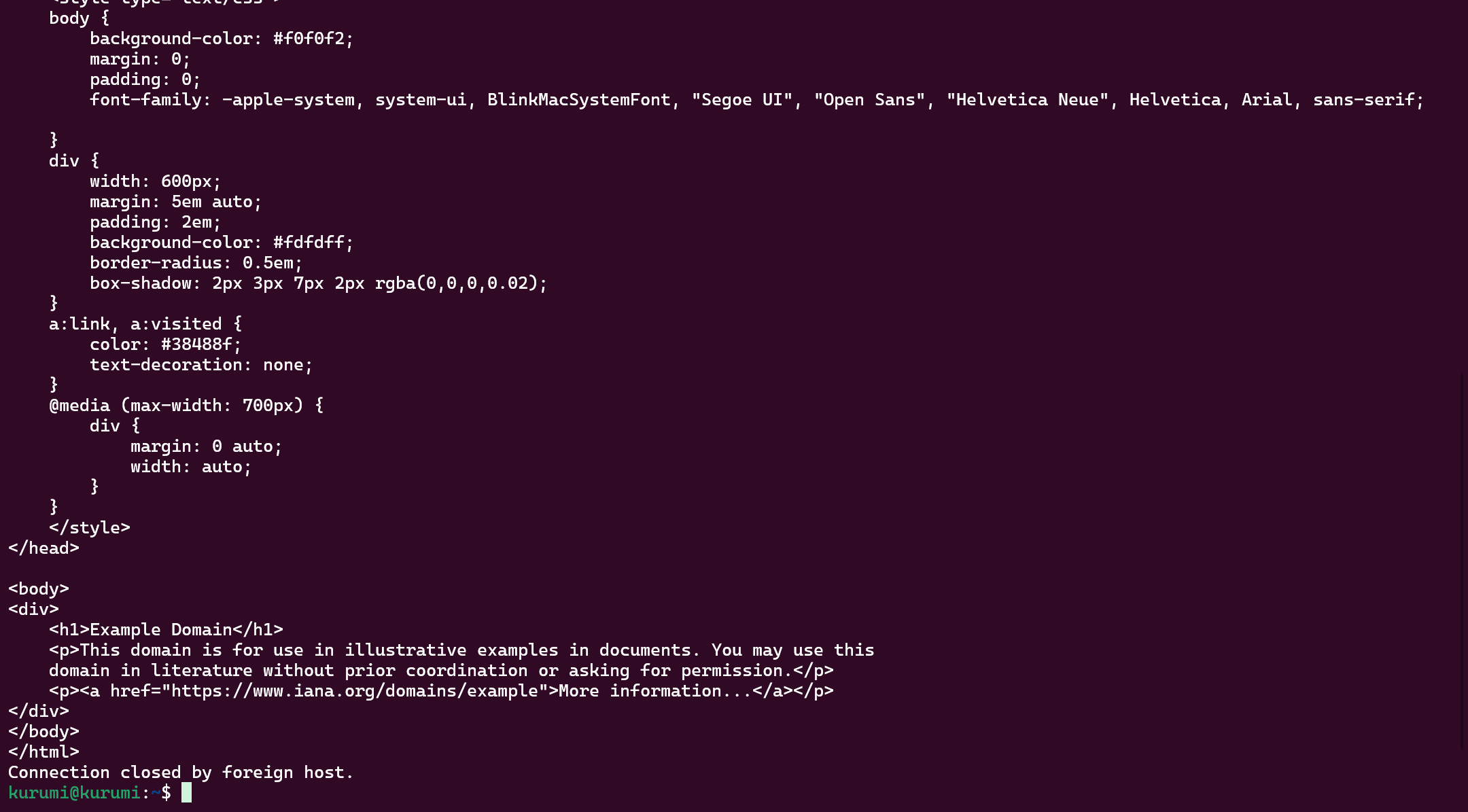
实验人：谢志康

学号：22307110187

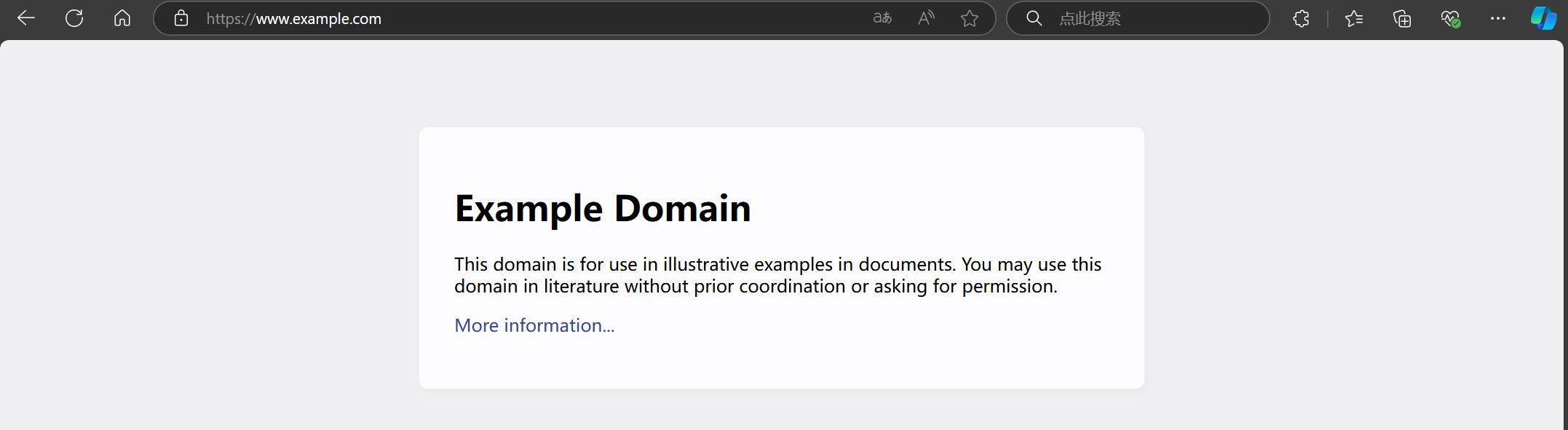
实验内容：应用层协议模拟与观察

**提交内容1**：进行以上操作和获取到的应答的命令行截图，并与直接使用浏览器访问该网页的内容进行对比。





可以看到HTTP/1.1 200 OK 表示服务器正常地应答了请求。浏览器打开example.com



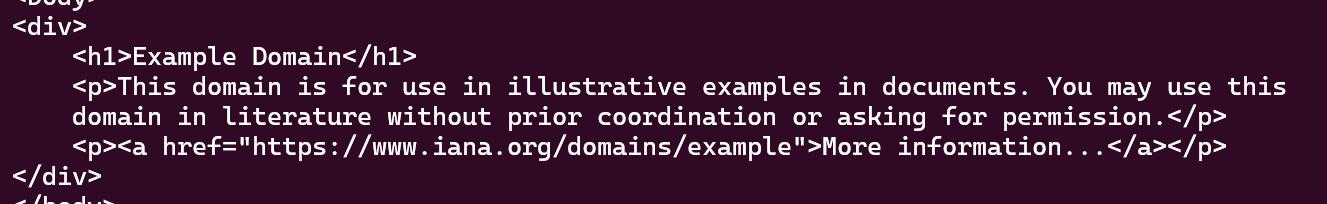
可以看到内容是一致的，我们请求获取到的信息和浏览器实际打开这个页面相同，获取到的是一个html文件：

<title>Example Domain</title> 定义这个页面在浏览器中显示的title



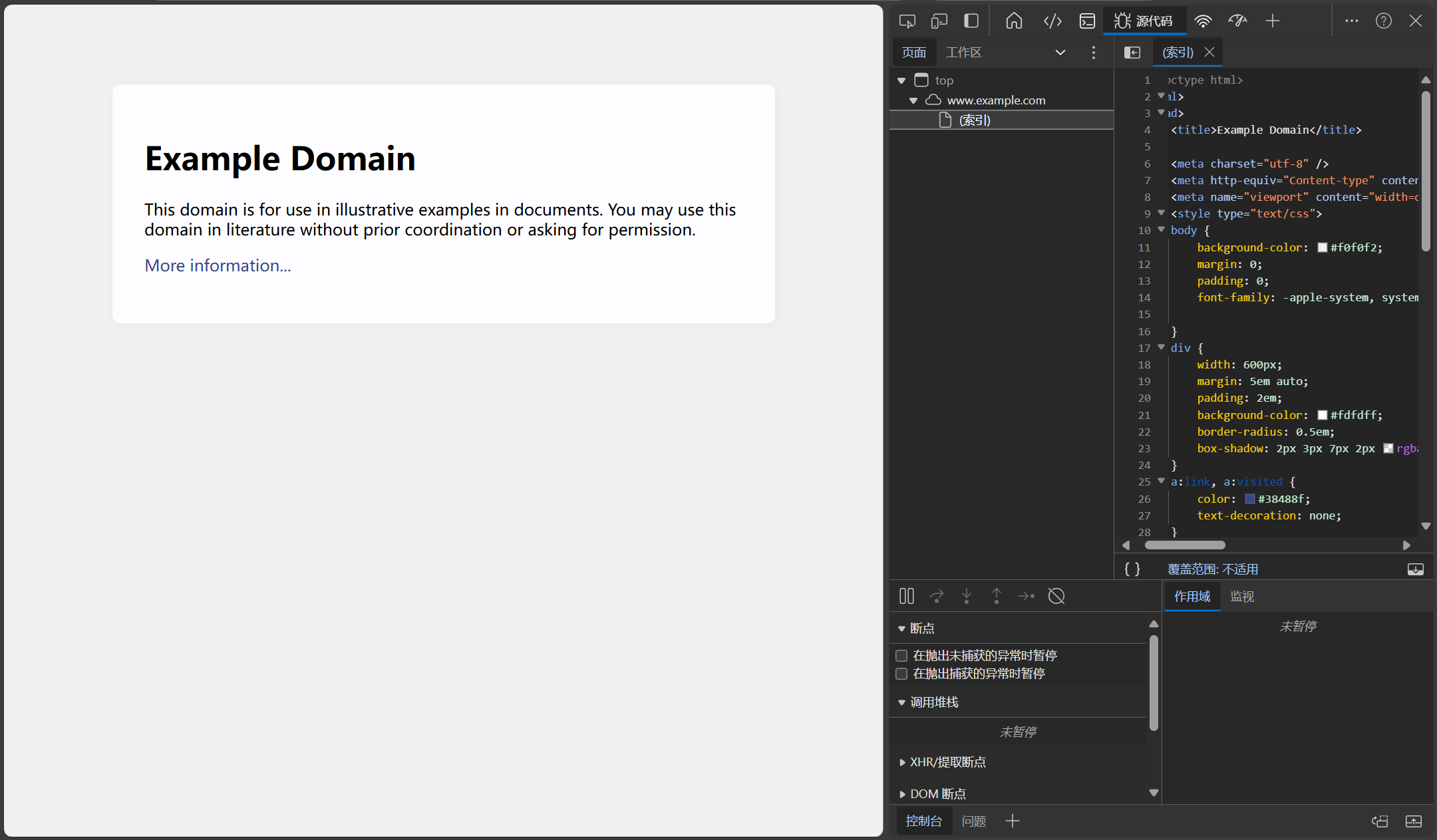
接下来一段只是定义了一下css样式

之后body部分就是内容，与浏览器上显示的一样



H1加粗的Example Domain字样，底下是<p>小字，href插入超链接，[www.iana.org](http://www.iana.org)......这个网站即是链接的目的网址，点进去发现确实是这个网站，More information是显示的字样。

进一步检查，f12打开浏览器的console，可以看到这个界面的html和我们telnet获取到的一模一样：

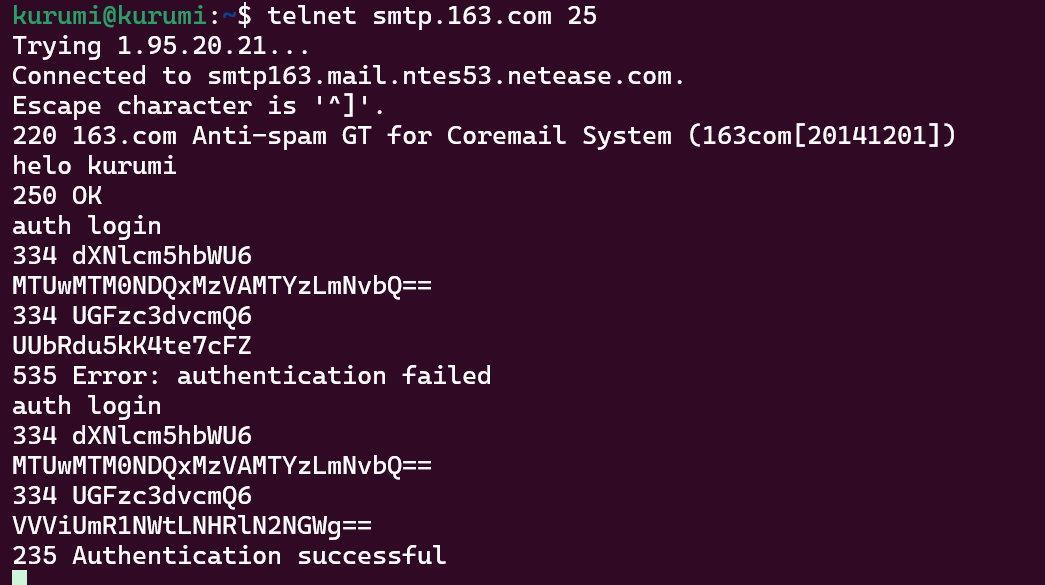


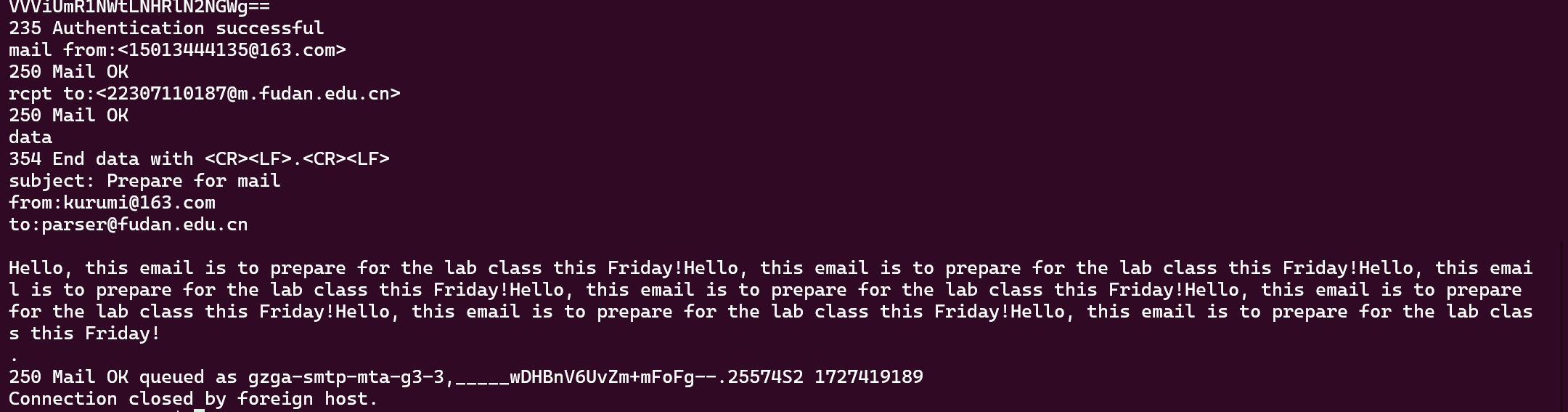
**提交内容2**：使用 telnet 发送邮件的过程截图，以及网页客户端中对方收到的邮件的截图。



首先网易邮箱要开启SMTP服务才行，不然登录报错550 User has no permission

然后网易会给一个仅展示一次的特殊密码，将其base64转换后登录即可。

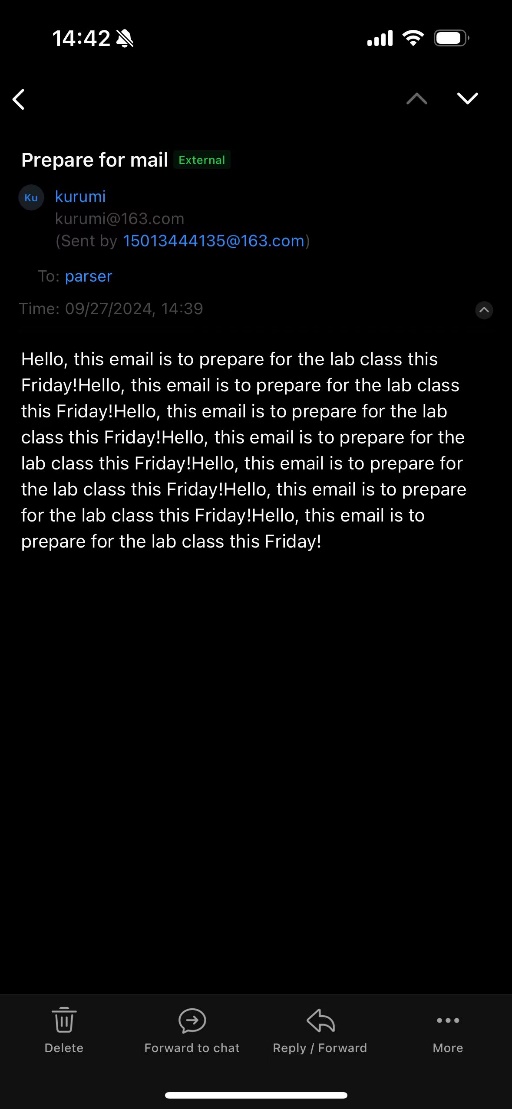






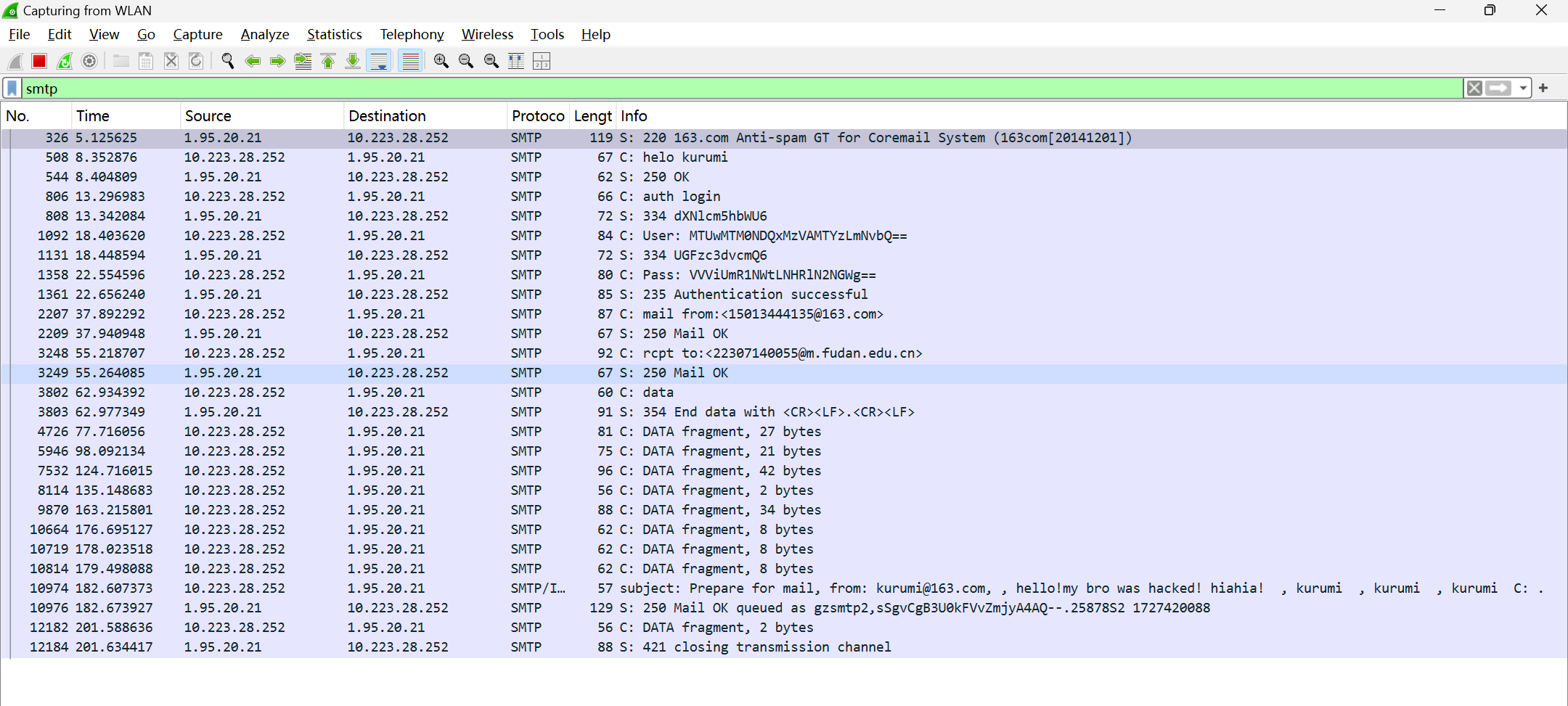
可以看到已发送邮件确实有刚刚发的邮件，但是真实发件人还是被detect出来，是我本人。

我另一个邮箱也确实收到了邮件。



**提交内容3：**在发送邮件的过程中，通过 Wireshark 抓包，截图并分析 SMTP 发送邮件的过程。

接下来从新发送一次，使用wireshark抓包：



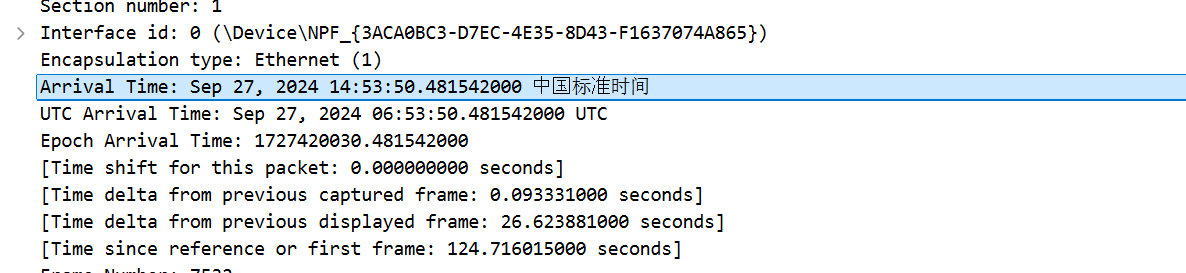
和我的操作步骤一样：

最开始建立连接，我输入helo kurumi，对面响应正常，回复250 OK

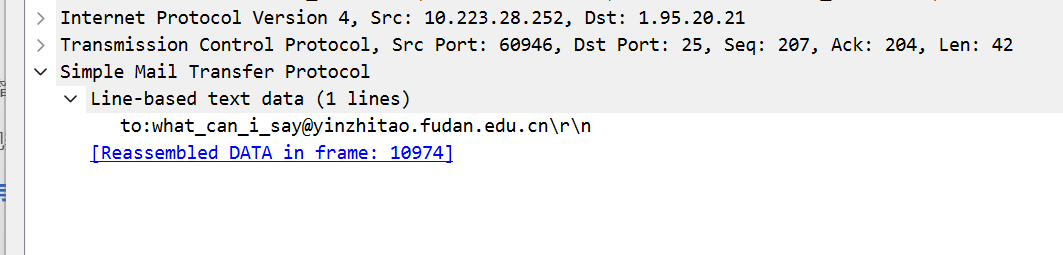
然后我登录，可以看到user和pass都是我输入的账号密码的base64编码，理论上只用base64解码即可获得明文账号密码，但是网易邮箱还是提供了保护机制的，我的实际密码登录它会显示no permission，开启SMTP服务后它会给我一个临时的密码才能SMTP登录，但是这个密码并不是我设置的实际密码，所以还是安全的。

返回登录成功后开始写邮件from和to是实际的发件人和收件人。后面的一段data fragment是我发件的信息，随便打开一个查看：

有这个data创建时间信息

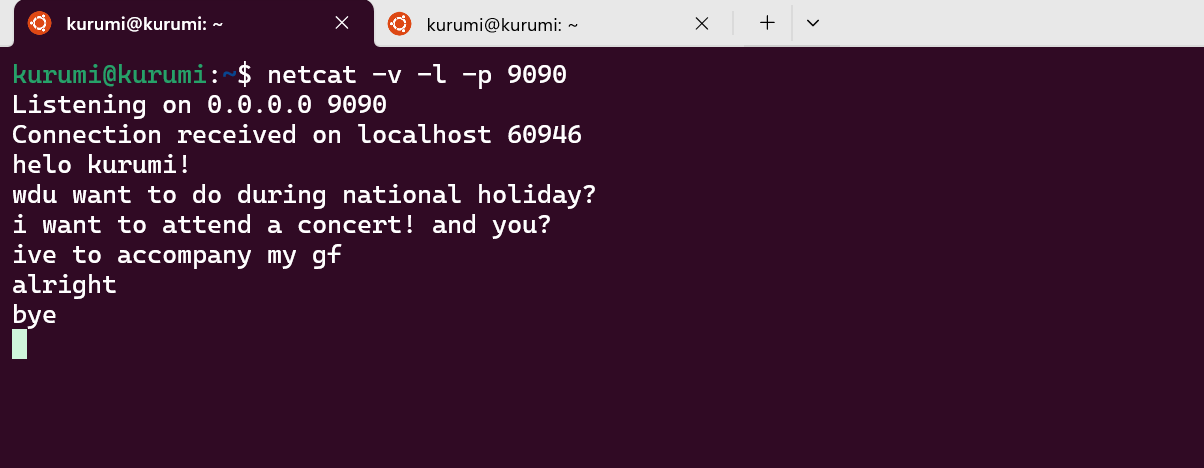


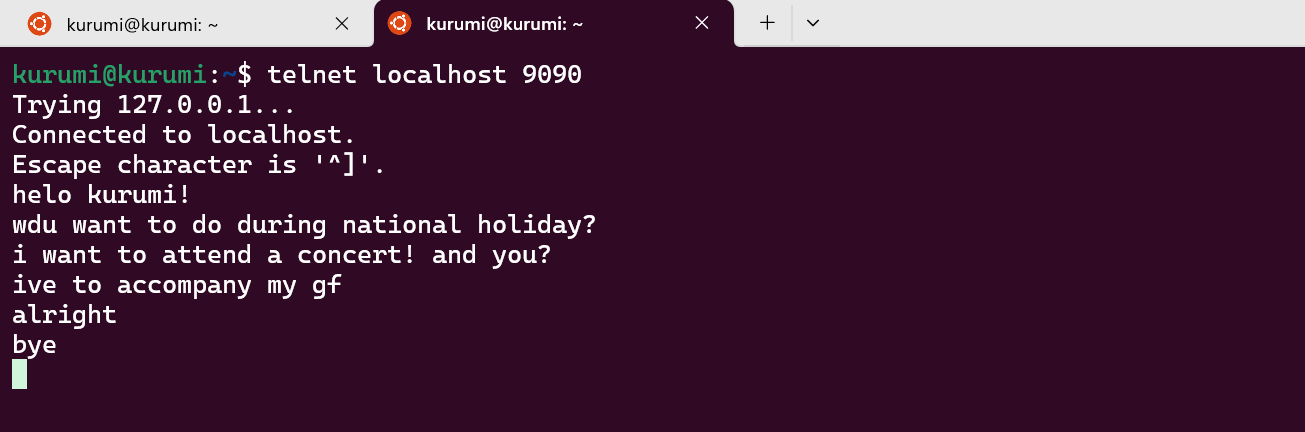
再看最后的SMTP：



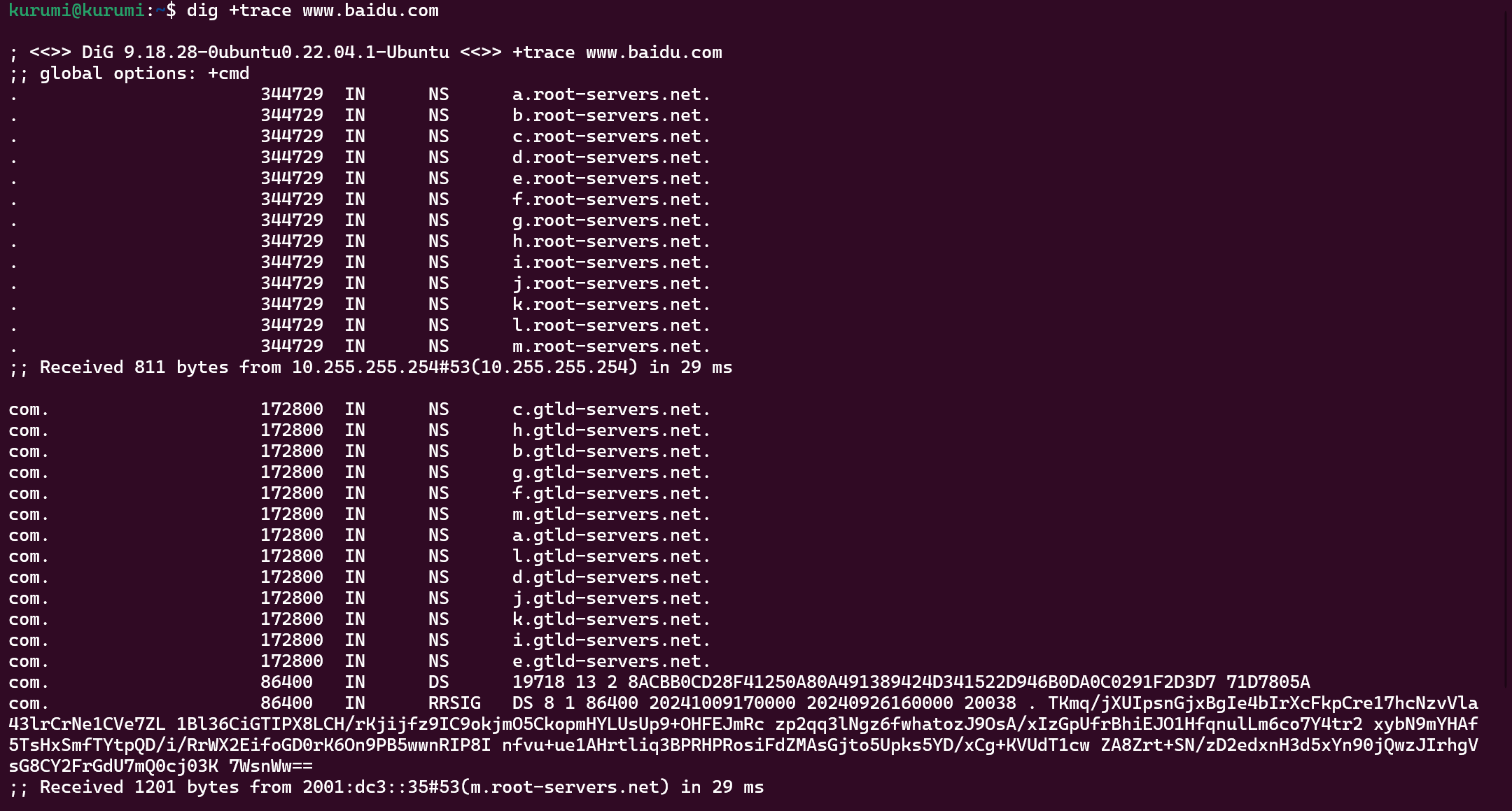
可以看到这一行我写的信件的真实内容。

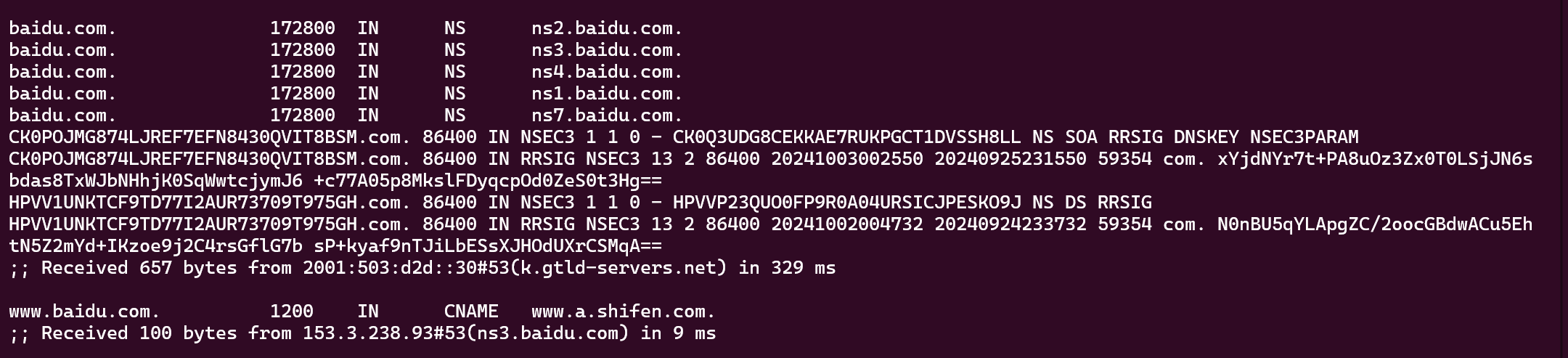
**提交内容4**：使用 Telnet 和 Netcat 模拟客户端和服务器进行信息传输的截图。





**提交内容5**：观察并分析 DNS 查询过程，判断查询采用的是递归查询还是迭代查询，并说明理由。





这里采用的是**迭代查询**。

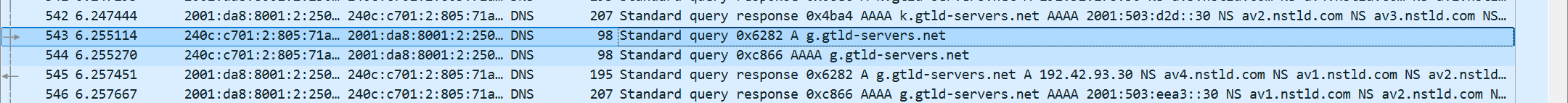
递归查询:服务器必需回答目标IP与域名的映射关系。要求 DNS 服务器处理整个查询过程，直到得到最终的结果。服务器需要进行多次查询，但不是将每一步都返回给客户端。

迭代查询:服务器收到一次迭代查询回复一次结果，这个结果不一定是目标IP与域名的映射关系，也可以是其它DNS服务器的地址。客户端与每个 DNS 服务器逐步交互，服务器只返回其知道的下一个 DNS 服务器地址，客户端负责继续查询。

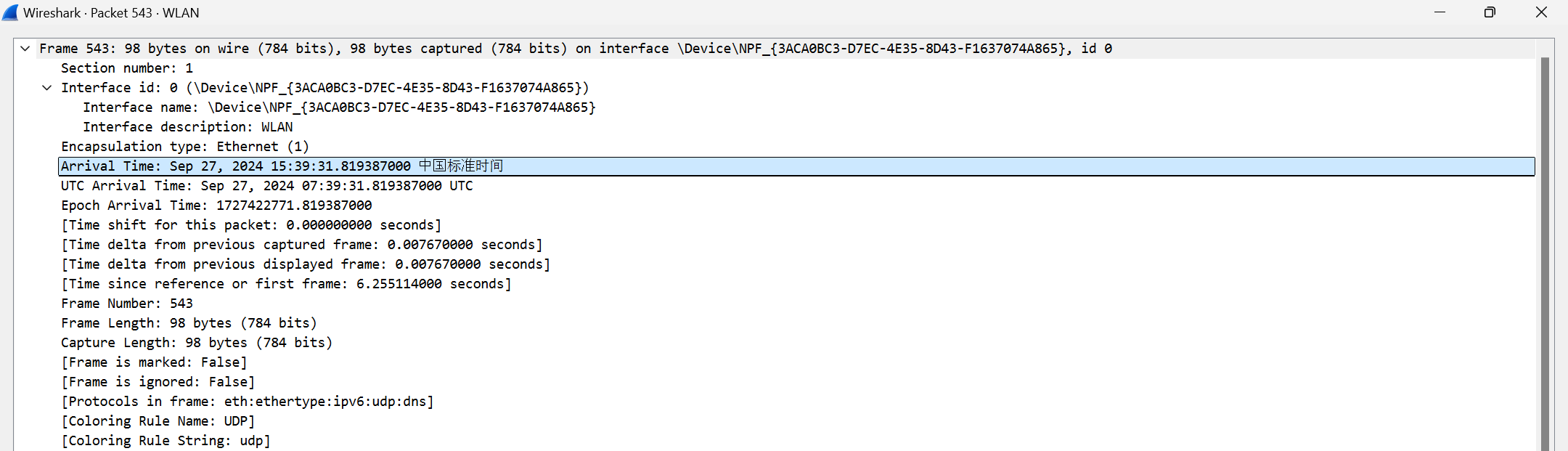
由截图可见，查询首先向根 DNS 服务器发送请求，根服务器返回负责 .com 域的顶级域名服务器的地址。由左边的.变为com. 然后查询继续向 .com 顶级域名服务器发送请求，获取负责 baidu.com 的域名服务器的地址。最后查询到达 baidu.com 的权威 DNS 服务器，获取到 www.baidu.com 的 CNAME 记录。

（同时，据我在网上了解到的知识，dig默认是递归方式查询，而使用 +trace 可以禁用递归查询转变为迭代查询。命令将显示查询的每一步过程，并提供每个服务器的响应。）

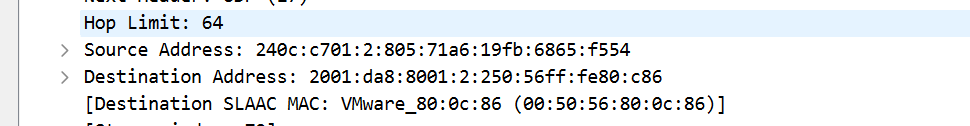
**提交内容6**：在 DNS 查询过程中使用 Wireshark 抓包，任选一对 DNS 请求与应答分组，解释 DNS 报文中各个字段的含义，以及标志（flag）字段中各个 flag 位的含义。

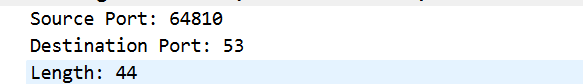


这是一对DNS请求与应答分组中的请求信息：时间，frame个数和大小都有标出

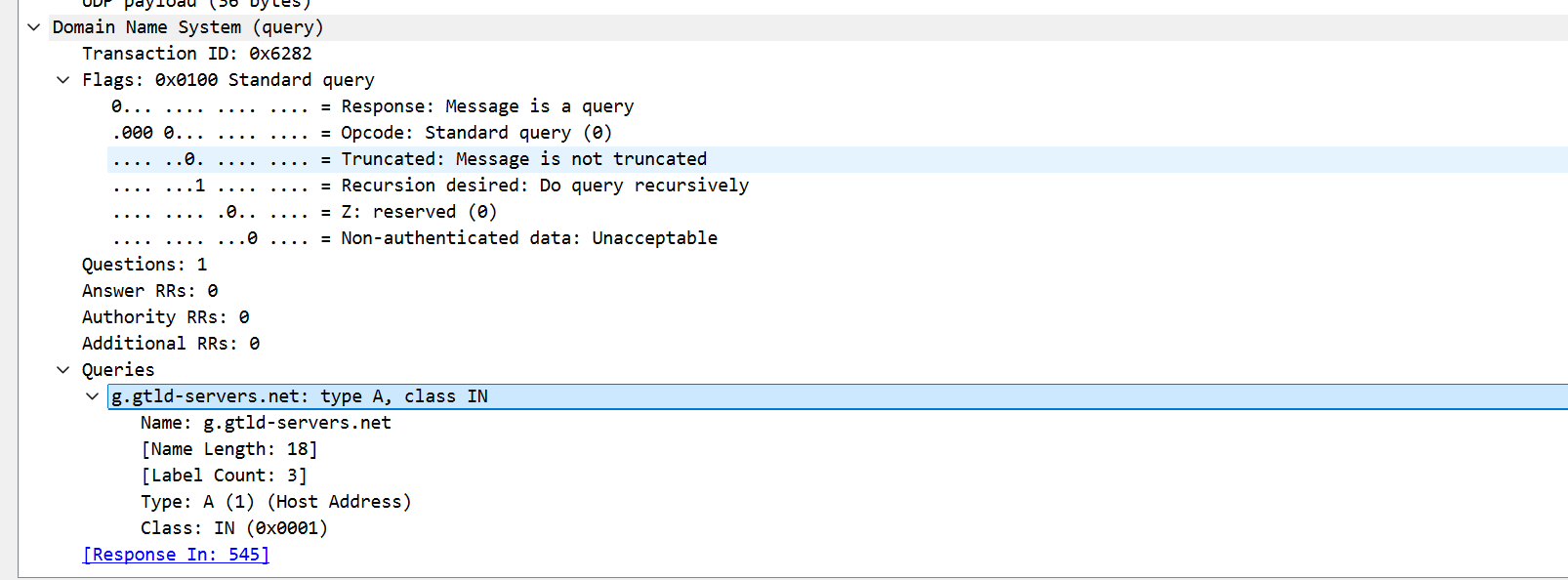


Source 和 destination的IP也有标出，端口号也有标出：





接下来是DNS信息：参考[DNS报文格式解析（非常详细）\_dns class in-CSDN博客](https://blog.csdn.net/anhenzhufeng/article/details/109860393)学习



Transaction ID是请求的ID，用于将请求与对应的响应匹配。 0x6282

Flags是0x0100

QR (0): Message is a query——表示这是一个查询（请求包）。

Opcode (0): 标准查询。

AA (0): 非权威应答。

TC (0): Message is not truncated——报文未被截断。

RD (1): 期望递归查询。

RA (0): 响应中不允许递归（在这个请求信息中无关）。

Z (0): reserved保留字段，必须为0。

AD (0): unacceptable——非认证数据。

CD (0): 不检查 DNSSEC 签名。

Questions:1 问题计数，这里查询的问题数为1

Answer RRs:0 没有答案资源记录（因为这是请求包）。

Authority RRs:0 权威名称服务器计数为0

Additional RRs:0 附加资源记录数为0

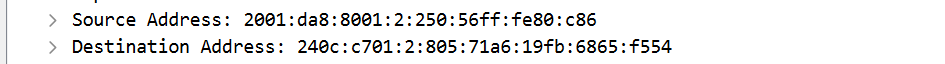
Name:g.gtld-servers.net 请求查询的域名。

Type A (1) (Host Address): 查询记录类型为 A（即 IPv4 地址）。

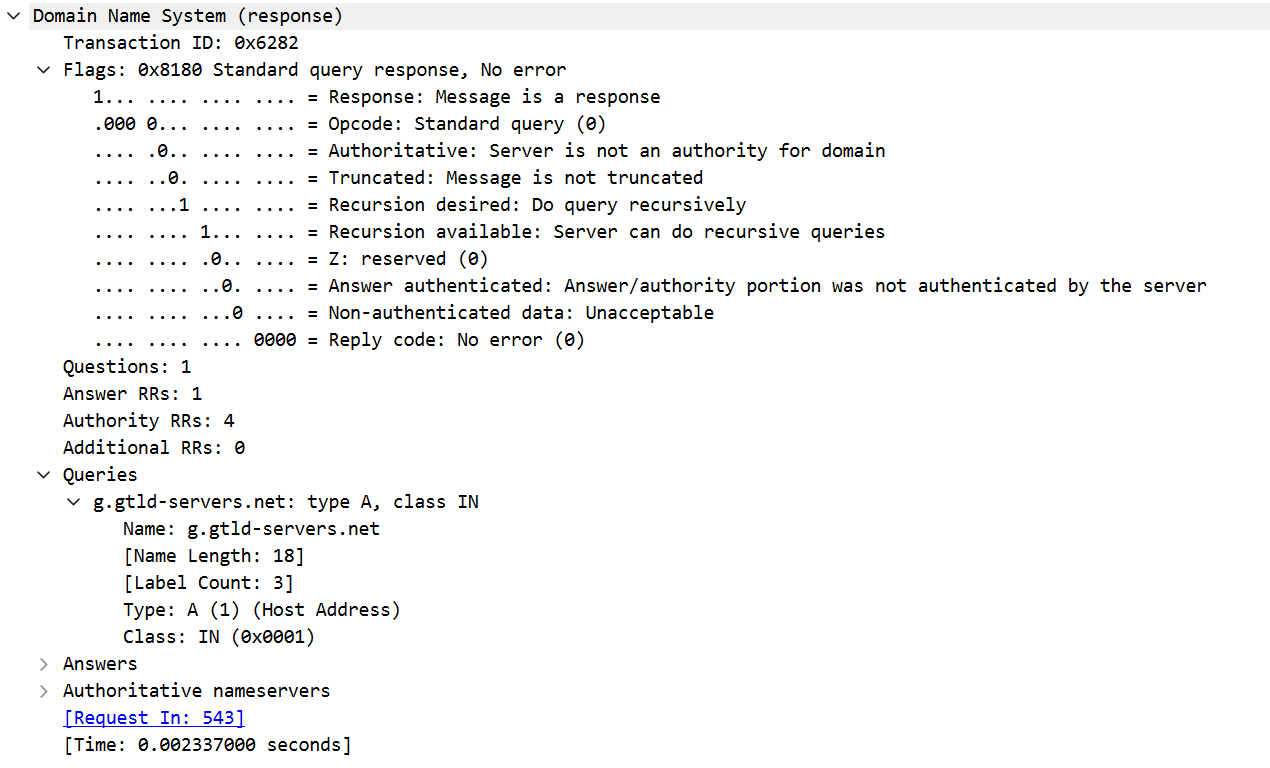
Class IN (0x0001): 表示查询的是互联网地址。

再看其对应的应答分组：

前面时间啊、frame大小等等信息，还有source和destination的IP地址和端口号等等也都有，不多赘述。



主要来看看DNS信息：



Transaction ID是0x6282，与它对应的请求包相同，以此来绑定。

Flags字段——

QR(1) Message is a response， 与前面QR(0)对应，这是一个响应包

后几个字段与请求包相同。Opcode (0): 标准查询响应。AA (0): 服务器不是权威服务器（非权威应答）。TC (0): 报文未被截断。RD (1): 递归查询被请求。多了个Recursion available：1 表示服务器支持递归查询。Z (0): 保留字段，必须为0。AD (0): 响应未被认证。CD (0): 检查 DNSSEC 签名。Reply code (0): no error没有错误，表示查询成功。

Questions (1): 查询的问题数为1。

Answer RRs (1): 1条答案资源记录，也就是找到一个回答。

Authority RRs (4): 有4条授权资源记录。

Additional RRs (0): 没有附加资源记录。

Queries: 这里与请求一致，查询的是 g.gtld-servers.net 的A记录（IPv4 地址）。Class IN(0x0001) 表示查询的是互联网地址。都与请求一致。

Answers: 响应包含 g.gtld-servers.net 的A记录。

Authoritative nameservers: 列出了授权名称服务器的记录，用于处理后续查询。

Response time表示响应时间为0.002337000秒。