

计网第五次作业

目录

一. 问题描述.....	1
二. 问题分析.....	2
三. 实验过程及代码.....	4
四. 结论.....	15
五. 参考文献.....	16

一.问题描述

Assignment 5: When the flow begins to flow~

1. use tools like wireshark and fiddler to tap and analyze the traffic when your PC browser,mobile phone browser and corresponding APP open a website such as taobao.cn,jd.com, pinduoduo.com and more (or any interesting websites/APPs), what versions of HTTP protocols do they use? Would they use different HTTP version for different scenarios?

2. illustrate the loading and rendering sequence of these websites, try explain their strategy to improve user experience 3. find out what information these website

gather from your PC and mobile phone. How do you feel? 4. Analyze the cryptographic systems used by these websites/APPs, what do they sign, verify, encrypt and decrypt?

任务 5: 当流开始流~

1. 使用 Wireshark、Fiddler 等工具, 在你的 PC 浏览器、手机浏览器和相应的 APP 打开淘宝、京东、拼多多等网站(或任何有趣的网站/APP)时, 对流量进行挖掘和分析, 他们使用的是哪个版本的 HTTP 协议? 他们会在不同的场景下使用不同的 HTTP 版本吗?

2. 说明这些网站的加载和渲染顺序, 尝试解释他们的策略, 以提高用户体验。

3. 看看这些网站从你的电脑和手机上收集了什么信息。你感觉怎么样?

4. 分析这些网站/应用程序使用的加密系统, 他们签署, 验证, 加密和解密什么?

二. 问题分析

本实验旨在通过多平台的流量捕获与解析, 揭示当前主流电商或热门网站在数据传输和安全保护上的技术细节。为此, 我们需要从以下几个方面展开分析:

HTTP 协议版本探测

现代网站可能同时支持 HTTP/1.1、HTTP/2 甚至 HTTP/3。借助抓包工具，可以从请求报文与响应报文中提取版本信息。不同平台（PC 浏览器、手机浏览器、APP 内置浏览器）由于底层网络库及优化策略的差异，可能在协议选择上存在差异。此外，不同请求场景（例如首页加载与后台数据请求）也可能使用不同的 HTTP 版本。

加载与渲染顺序策略

网站加载流程通常包括 DNS 解析、TCP/UDP 连接建立、TLS 握手（若为 HTTPS）、发送 HTTP 请求、接收响应、页面资源加载、CSS 渲染与 JavaScript 执行。网站为了提高页面响应速度，往往会采用资源预加载、异步加载、延迟渲染等优化策略，本实验将结合抓包数据和浏览器开发者工具观察页面的加载顺序与时间分布。

用户信息收集

网站和 APP 通常会通过 Cookie、LocalStorage、设备指纹等方式收集用户信息，包括硬件设备、操作系统、浏览器类型、地理位置信息以及用户行为数据。通过分析 HTTP 报文和 APP 通信数据，能够探查出这些信息采集的细节，并就其隐私保护与安全性进行讨论。

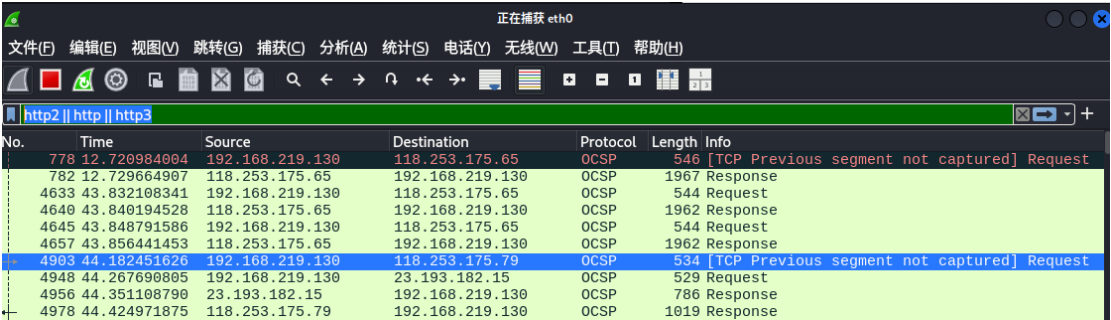
加密系统与安全机制

在 HTTPS 环境下，数据传输采用 SSL/TLS 加密。我们将重点关注证书签署、握手流程以及数据加密传输过程，分析网站如何利用加密机制确保数据完整性和

机密性，同时探讨其在身份验证和防篡改方面的作用。

三.实验过程及代码

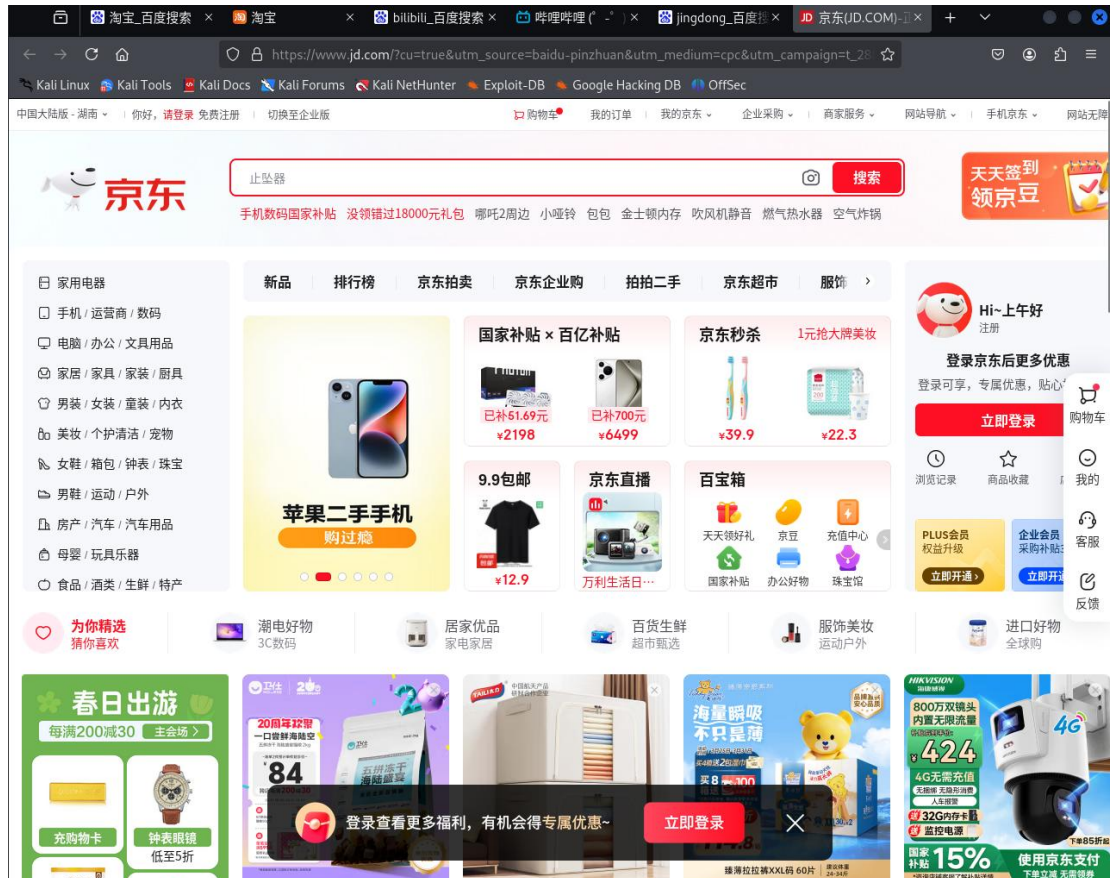
首先，我们在 kali 中打开软件 wireshark，因为要查看 http 协议，所以我们在软件顶部的过滤器中输入 http2 || http || http3 来进行过滤，使得软件只显示这些 http 协议相关内容：



The image shows the Wireshark interface with the filter 'http2 || http || http3' applied. The packet list table is as follows:

No.	Time	Source	Destination	Protocol	Length	Info
778	12.720984004	192.168.219.130	118.253.175.65	OCSP	546	[TCP Previous segment not captured] Request
782	12.729664907	118.253.175.65	192.168.219.130	OCSP	1967	Response
4633	43.832108341	192.168.219.130	118.253.175.65	OCSP	544	Request
4640	43.840194528	118.253.175.65	192.168.219.130	OCSP	1962	Response
4645	43.848791586	192.168.219.130	118.253.175.65	OCSP	544	Request
4657	43.856441453	118.253.175.65	192.168.219.130	OCSP	1962	Response
4983	44.182451626	192.168.219.130	118.253.175.79	OCSP	534	[TCP Previous segment not captured] Request
4948	44.267690805	192.168.219.130	23.193.182.15	OCSP	529	Request
4956	44.351108790	23.193.182.15	192.168.219.130	OCSP	786	Response
4978	44.424971875	118.253.175.79	192.168.219.130	OCSP	1019	Response

然后，我们打开网站：



打开之后，我们就能在 wireshark 中看到相关的内容了。

11156	65.586807915	192.168.219.130	183.255.35.38	TCP	54	33162 → 443	[ACK] Seq=3854 Ack=11677806 Win=65535 Len=0
11157	65.586854889	183.255.35.38	192.168.219.130	TLSv1.2	33634	Application Data, Application Data	
11158	65.58686429	192.168.219.130	183.255.35.38	TCP	54	33162 → 443	[ACK] Seq=3854 Ack=11711386 Win=65535 Len=0
11159	65.586867698	183.255.35.38	192.168.219.130	TLSv1.2	11734	Application Data	
11160	65.586881943	192.168.219.130	183.255.35.38	TCP	54	33162 → 443	[ACK] Seq=3854 Ack=11723066 Win=65535 Len=0
11161	65.586914285	183.255.35.38	192.168.219.130	TCP	10274	443 → 33162	[ACK] Seq=11723066 Ack=3854 Win=64240 Len=10220 [TCP PDU reassembled in 443853]
11162	65.586917624	192.168.219.130	183.255.35.38	TCP	54	33162 → 443	[ACK] Seq=3854 Ack=11733286 Win=65535 Len=0
11163	65.586958206	183.255.35.38	192.168.219.130	TLSv1.2	42393	Application Data, Application Data, Application Data, Application Data	
11164	65.586966669	192.168.219.130	183.255.35.38	TCP	54	33162 → 443	[ACK] Seq=3854 Ack=11775625 Win=65535 Len=0
11165	65.586969614	183.255.35.38	192.168.219.130	TLSv1.2	11734	Application Data	
11166	65.586990585	192.168.219.130	183.255.35.38	TCP	54	33162 → 443	[ACK] Seq=3854 Ack=11787385 Win=65535 Len=0
11167	65.587021592	183.255.35.38	192.168.219.130	TCP	8814	443 → 33162	[ACK] Seq=11787385 Ack=3854 Win=64240 Len=8760 [TCP PDU reassembled in 443853]
11168	65.587025877	192.168.219.130	183.255.35.38	TCP	54	33162 → 443	[ACK] Seq=3854 Ack=11796065 Win=65535 Len=0
11169	65.587235739	183.255.35.38	192.168.219.130	TLSv1.2	43853	Application Data, Application Data, Application Data, Application Data	

这里我们看到 wireshark 中是没有 http 或类似的协议展示的，经过网上查询相关内容之后，我们知道了，这是因为如今大多数网站都使用 HTTPS，而不是明文的 HTTP。HTTPS 在传输过程中会使用 TLS/SSL 对数据进行加密，因此在 Wireshark 中捕获到的往往显示为 TLS，而不是 HTTP。如果你没有进行解密，就无法直接看到 HTTP 报文。虽然我们没看到 http，但是我们可以看到其中有很多的 tls 协议，所以我们要调整设置来进行对其的解密，我们可以使用设置环境变量密钥日志文件来实现：配置浏览器生成密钥日志文件在终端中，先创建一个存放密钥日志的文件，例如：mkdir -p /tmp/sslkeys touch /tmp/sslkeys/sslkeylog.log

然后再设置环境变量，使浏览器在启动时将 TLS 会话密钥记录到该文件。以 Firefox 为例，可在终端中执行：`export SSLKEYLOGFILE=/tmp/sslkeys/sslkeylog.log` `firefox &`

如果使用 Chrome，则类似：`export SSLKEYLOGFILE=/tmp/sslkeys/sslkeylog.log`
`google-chrome &`

注意：确保在同一个终端中设置环境变量后再启动浏览器，这样浏览器就会写入密钥到指定文件中。



```
(kali㉿kali)-[~/桌面]
$ mkdir -p /tmp/sslkeys

(kali㉿kali)-[~/桌面]
$ touch /tmp/sslkeys/sslkeylog.log

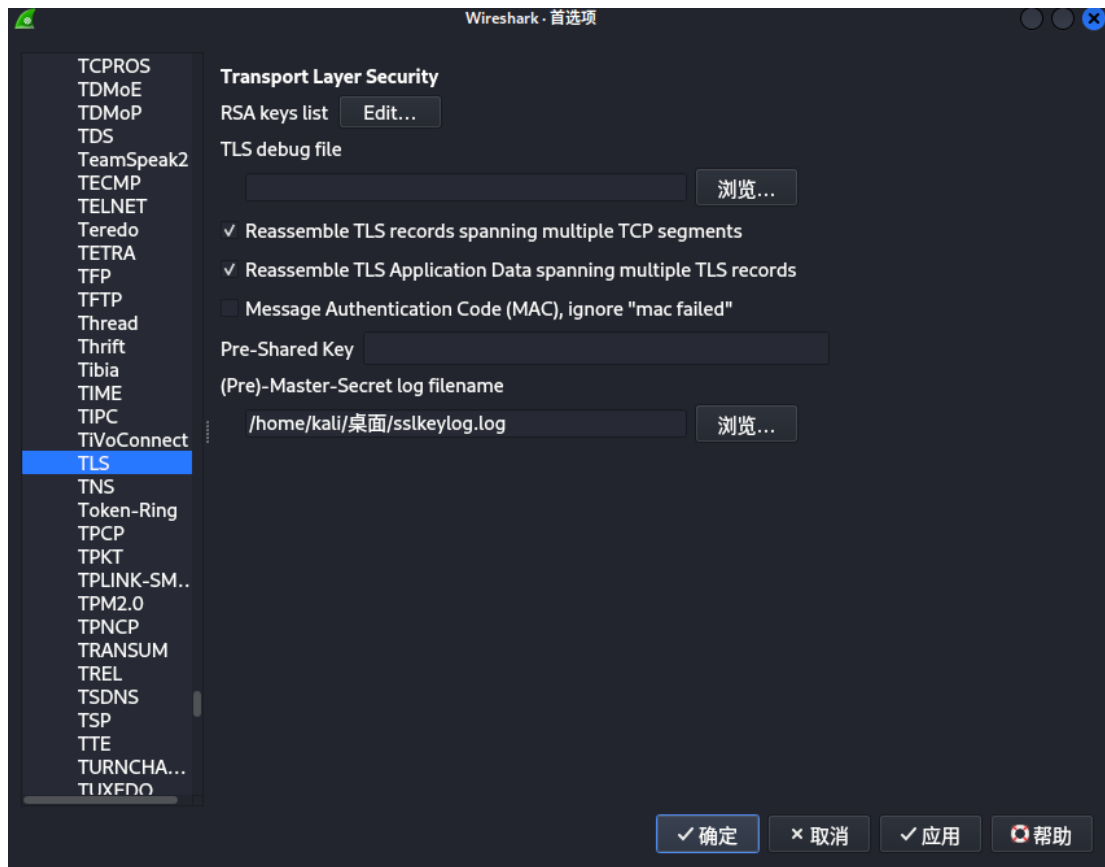
(kali㉿kali)-[~/桌面]
$ export SSLKEYLOGFILE=/tmp/sslkeys/sslkeylog.log

(kali㉿kali)-[~/桌面]
$ firefox &
[1] 35232

(kali㉿kali)-[~/桌面]
$
[1] + done firefox

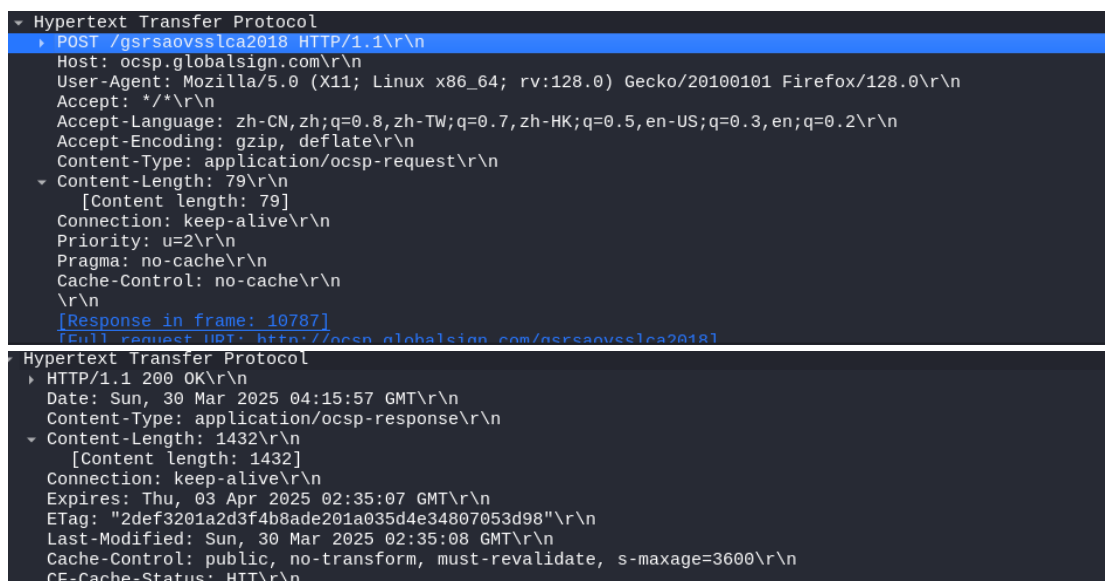
(kali㉿kali)-[~/桌面]
$
```

生成了密钥文件之后，我们就打开 Wireshark 进行配置：在 Wireshark 中指定密钥日志文件打开 Wireshark，点击【Edit】→【Preferences】（首选项）在左侧面板中展开【Protocols】，找到【TLS】在“(Pre)-Master-Secret log filename”字段中填写密钥日志文件路径：`/tmp/sslkeys/sslkeylog.log` 点击【OK】保存设置，并重启 Wireshark 或开始新的捕获。



此时，捕获的 TLS 流量（HTTPS）会使用该日志文件中的密钥进行解密，显示出明文的 HTTP 信息。

我们可以选取其中几条记录来进行查看：



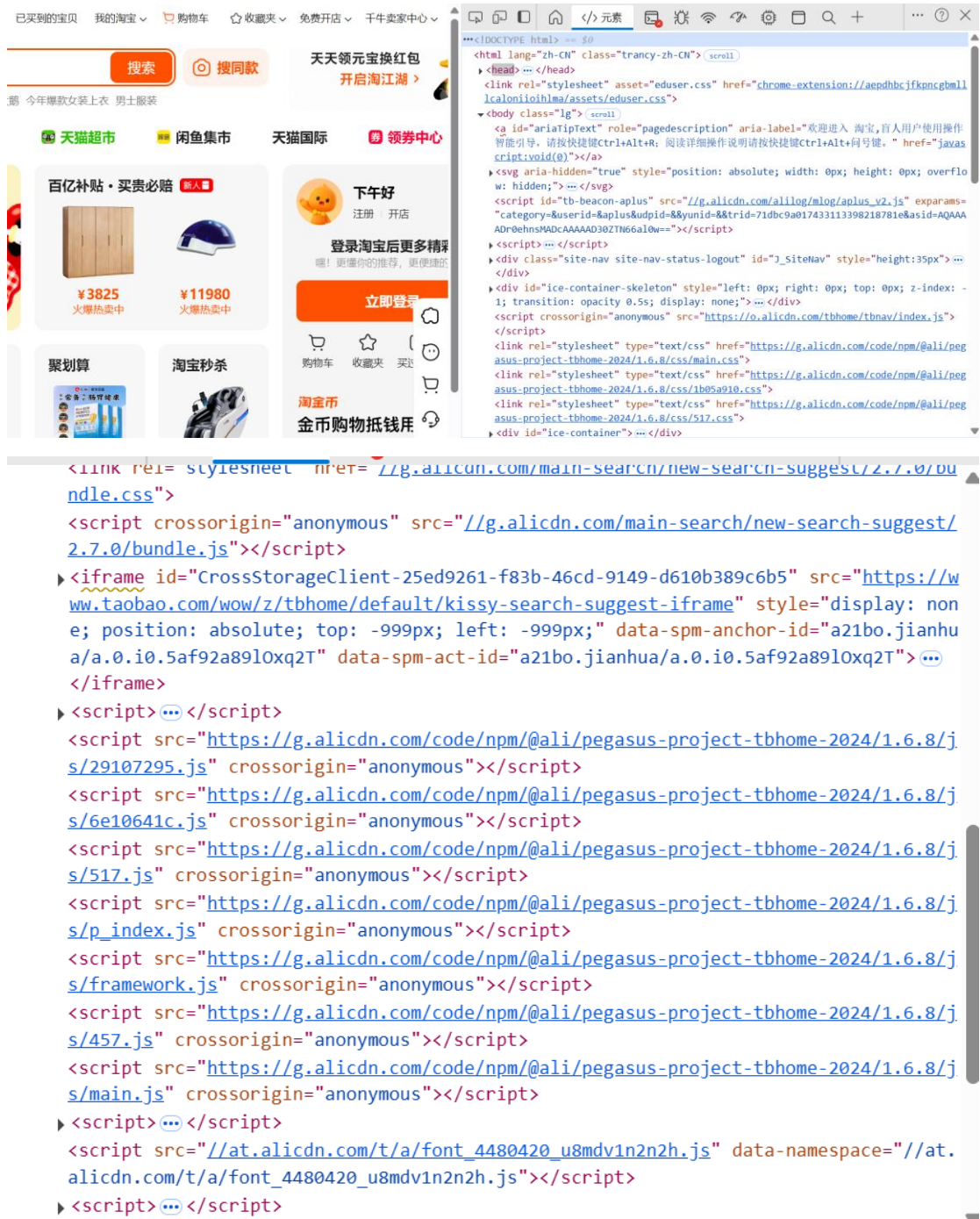
```
HTTP/1.1 200 OK\r\n
Date: Sun, 30 Mar 2025 04:15:54 GMT\r\n
Content-Type: application/ocsp-response\r\n
Content-Length: 1432\r\n
[Content length: 1432]
Connection: keep-alive\r\n
Expires: Thu, 03 Apr 2025 02:35:07 GMT\r\n
ETag: "2def3201a2d3f4b8ade201a035d4e34807053d98"\r\n
Last-Modified: Sun, 30 Mar 2025 02:35:08 GMT\r\n
Cache-Control: public, no-transform, must-revalidate, s-maxage=3600\r\n
CF-Cache-Status: HIT\r\n
Age: 2266\r\n
```

可以看到大部分都是 http/1.1。其中有 post, 也有 ok 等回复。在网上搜集资料, 看看为什么大部分都会选择使用 http1.1 版本。许多网站依然选择使用 HTTP/1.1, 主要原因包括以下几个方面:

- 1 广泛兼容性 HTTP/1.1 已经存在多年, 几乎所有服务器、代理、负载均衡器以及安全设备都对它有很好的支持。对于许多网站来说, 使用 HTTP/1.1 可以确保与各种客户端和中间设备的兼容性, 减少因协议转换引起的问题。
- 2 部署和维护成本 虽然 HTTP/2 和 HTTP/3 在性能和并发性上有优势, 但升级服务器和相关网络设备、修改应用架构都需要一定的投入和技术改造。对于一些网站来说, 短期内继续使用 HTTP/1.1 可以降低成本和风险。
- 3 实际收益考虑 对于内容量较小或用户访问压力不高的网站, HTTP/1.1 在实际使用中已经足够满足需求。只有在需要大规模并发和更高传输效率时, HTTP/2 或 HTTP/3 的优势才会更明显。
- 4 渐进式迁移策略 许多网站采取渐进式升级策略, 在某些特定场景下可能会部分启用 HTTP/2, 但整体仍以 HTTP/1.1 为主, 这样可以在保持现有系统稳定性的同时逐步过渡到新协议。

总的来说, 虽然新协议在性能上有明显改进, 但兼容性、成本和稳定性等因素使得很多网站在当前阶段依然选择继续使用 HTTP/1.1。

接下来, 我们要找到网站的渲染和加载顺序, 这点直接在浏览器中查看就行。我们打开淘宝的网站, 并对此进行观测:



```

<script src="https://g.alicdn.com/code/npm/@ali/pegasus-project-tbhome-2024/1.6.8/js/framework.js" crossorigin="anonymous"></script>
<script src="https://g.alicdn.com/code/npm/@ali/pegasus-project-tbhome-2024/1.6.8/js/457.js" crossorigin="anonymous"></script>
<script src="https://g.alicdn.com/code/npm/@ali/pegasus-project-tbhome-2024/1.6.8/js/main.js" crossorigin="anonymous"></script>
<script></script>
<script src="//at.alicdn.com/t/a/font_4480420_u8mdv1n2n2h.js" data-namespace="//at.alicdn.com/t/a/font_4480420_u8mdv1n2n2h.js"></script>
<script></script>
<script crossorigin="anonymous" src="https://o.alicdn.com/tbpc/securitySDK/securitySDK.umd.js"></script>
<script></script>
<script></script>
<div style="width:100%;background-color:var(--tbpc-white-color, #fff);margin-top:20px;overflow:unset" class="tbpc-layout" data-spm="seoinject"></div>
<div id="J_SiteFooter" style="min-height: 170px;"></div>
<div id="J_TBPC_POP_home"></div>
<script src="//g.alicdn.com/dinamic/barrier-free/0.0.14/aria.js?appid=7e39dd4..." charset="utf-8" crossorigin="anonymous" id="ariascripts"></script>
<div id="J_Toolkit"></div>
</body>
</html>

```

我们可以对这些元素进行分析，通过浏览器自带的开发者工具和抓包数据，我们进一步分析了页面加载和渲染的详细过程：

① DNS 解析与连接建立：首先进行 DNS 查询，将域名解析为 IP 地址；紧接着建立 TCP 连接，并进行 TLS 握手（若为 HTTPS）；

② 页面骨架请求：加载 HTML 主文档后，浏览器解析 DOM 树，并逐步发送对 CSS、JavaScript、图片等资源的请求；

③ 资源并行加载：多数网站采用资源合并、异步加载与延迟加载策略，部分静态资源（如广告、跟踪脚本）会在页面主体加载完成后再进行请求；

④ 渲染与回流：浏览器根据 CSS 规则渲染页面，执行 JavaScript 进行动态效果呈现，部分操作（如图片懒加载）则基于滚动事件触发。



1 是加快页面响应

通过预加载和懒加载，用户可以更快地看到页面的主要内容，而不必等待所有资源下载完毕。同时异步加载避免了阻塞，确保页面交互及时响应。

2 是提升视觉和交互效果

-通过合理的资源管理和优化，多媒体内容和动态效果可以在不拖慢页面加载的情况下呈现，为用户带来流畅的视觉体验。渐进式渲染和页面骨架屏技术让用户在等待过程中也能感受到页面正在加载，减少等待焦虑。

3 是减少带宽与服务器负荷

资源压缩、合并及缓存策略不仅加速加载，也降低了对网络带宽和服务器资源的消耗，进一步提高整体性能。CDN 的应用确保即使在全球范围内访问，也能获得较快的响应速度。

4 是适应不同网络环境和终端

移动端和桌面端可能存在网络速度和硬件性能上的差异，采用响应式设计和动态资源加载策略可以在不同终端上都提供良好体验。针对低带宽环境的优化，如懒加载和压缩技术，保证即使网络较差也能流畅浏览。

总的来说，网站通过精心设计页面结构、优化资源加载顺序、利用异步和懒加载技术、结合 CDN 与缓存策略等多种手段，不仅提高了页面加载速度和交互响应，还在视觉效果和整体稳定性上显著提升了用户体验。这些策略相辅相成，使得现代网站在不断追求速度和美观的同时，也能适应多变的网络环境和用户需求。

下一个任务：查看网站从我们这里获取到的信息。我们找到淘宝官方公开的隐私政策。：

二、信息收集及使用

为实现向您提供我们产品及/或服务的基本功能，您须授权我们收集、使用的必要的信息。**如您拒绝提供相应信息，您将无法正常使用我们的产品及/或服务。**

我们会为您提供的各项具体功能场景包括：

（一）帮助您成为我们的会员及账户管理

1、淘宝平台账户服务

我们基于淘宝平台账户为您提供服务。为了创建淘宝平台账户（又称“淘宝平台账号”或“淘宝账号”）您需要至少向我们提供您的**手机号码、拟使用的会员名和密码**。如果您拒绝提供上述信息，您将无法注册淘宝平台账户，仅可以使用浏览、搜索服务。我们会根据您的淘宝平台账户使用情况及平台设置的账户等级计算规则确定您当前的等级，并为您提供相应账户等级所对应的基本权益。

您可通过淘宝平台账户在我们网站或者客户端提供的链接入口或其他产品/服务入口使用我们及/或关联公司提供的产品或服务。为方便您管理淘宝平台账户，我们会在淘宝平台账户服务中同步并显示您的注册信息，以及您在淘宝平台上或与淘宝平台账户相关联的产品和服务中执行的操作，包括通过淘宝平台账户集中展示您的个人资料、优惠权益、交易订单。您淘宝平台账户所关联的上述信息，将会被用于为您提供客户服务，识别账号安全风险，以及推荐您可能感兴趣的商品或服务信息。

您可自行设置昵称、性别和头像，这些信息将在评价、分享及其他互动场景中公开显示。您可自行设置您的出生年月日信息，淘宝可能根据您的生日信息为您展示生日相关权益。该类信息属于非必要个人信息，您不提供前述信息不会影响您使用淘宝平台服务。

2、授权登录

我们可能经您单独同意后向第三方提供您的账户信息（头像、昵称及其他页面提示的信息），使您可以便捷地实现第三方账户的注册或第三方处直接登录。此外，我们可能会根据您的单独同意从第三方处获取您的第三方账户信息，并与您的淘宝平台账户进行绑定，使您可在第三方处直接登录、使用我们的产品及/或服务。我们将在您单独同意的授权范围内使用您的相关信息。

3、身份验证

在其中的信息收集及使用条例中，我们能找出其从我们这里获取到的信息大致如下：

手机号码、拟使用的会员名和密码。

从第三方处获取您的第三方账户信息

相关信息进行身份验证

实名信息

淘宝账号绑定手机号的风险情况

相关真实身份信息（姓名、证件号码等证件信息或面部识别信息）

操作记录

搜索记录

麦克风权限、相机权限或您提供的图片

访问或使用淘宝平台网站或客户端时的浏览、搜索、加购、交易记录并结合依法收集的设备信息、服务日志信息

.....

可以看到淘宝从我们这里获取了许多的信息以及权限。其中有部分是为了法律于安全考虑，比如强制的实名以及手机号码的风险评估，还有部分是为了更好地提供服务，比如根据搜索记录更精准地投送用户可能需要的物品。

总的来讲，有利有弊，如果这些公开信息没有被用在其他超出范围的地方的话，我本人对此意见并不大。

下一个任务，查看平台的安全与加密解密情况。我们去网上搜寻答案。淘宝的加密系统主要基于 HTTPS/TLS 协议来保护数据传输安全，其核心过程包括数字证书的签署、验证以及密钥交换后对数据进行对称加密和解密。

签署：淘宝的服务器会持有由受信任的证书颁发机构（CA）签发的数字证书。在这个过程中，CA 使用其私钥对服务器证书的摘要进行签名，从而保证证书内容（包括服务器的公钥、身份信息和有效期等）的完整性和真实性。

验证：当客户端（如浏览器或 APP）与淘宝建立 HTTPS 连接时，服务器会发送自己的数字证书。客户端利用预先安装的受信任 CA 列表中的公钥来验证该证书：检查证书签名是否正确，即证书是否确实由信任的 CA 签署；确认证书中的域名与所访问的域名匹配；检查证书是否在有效期内。这一验证过程确保客户端与真实的淘宝服务器建立安全连接。

加密：在 TLS 握手阶段，客户端和服务器通过协商确定一组加密算法和密钥交换方法。常见的方案包括采用 RSA 或（更常见的）椭圆曲线 Diffie-Hellman

(ECDHE) 进行密钥交换：客户端生成一个预主密钥，并使用服务器的公钥将其加密后发送给服务器；双方利用预主密钥和握手过程中交换的随机数，生成相同的对称加密密钥。此对称密钥随后用于加密实际的数据传输，确保数据在网络上传输时不被窃听。

解密：数据在传输过程中由双方使用相同的对称密钥进行加密。接收方利用该密钥将收到的加密数据解密，还原为明文数据。这样，无论是请求数据还是响应数据，双方都能确保通信内容的机密性和完整性。

此外，淘宝可能会采用 TLS 1.2 或 TLS 1.3 等较新版本的协议，结合 ECDHE 等前向保密（Forward Secrecy）技术，确保即便服务器私钥在未来泄露，之前的通信内容依然安全。整体来看，淘宝的加密系统通过数字证书的签署与验证、密钥交换过程中的加密和对称加密算法的应用，实现了从身份验证到数据加解密的全链路保护，保障用户数据传输的安全性。

四.结论

我们在这次实验中通过对 PCP 端访问淘宝、京东、拼多多等平台的流量分析，得出以下主要结论：

- 1 是 HTTP 协议版本问题：各平台普遍采用 HTTP/1.1 作为默认协议。APP 端由于使用定制库，协议版本显示出一定的多样性；
- 2 是加载与渲染的策略：目标网站普遍采取分步加载、资源异步调度及首屏优先策略，通过优化 DNS、TCP/TLS 握手、静态资源合并等手段，提高页面加载速度

和用户体验;

3 是信息采集现状: 网站和 APP 通过多种方式收集用户设备与行为信息, 这既有助于服务优化, 也引发了隐私保护方面的担忧。我认为用户和平台均需在体验与隐私之间找到平衡;

4 是加密系统分析: 各目标平台在 HTTPS 传输中均采用了成熟的加密技术和证书验证机制, 保证了数据传输的安全性, 但同时也需关注新型攻击方式, 持续更新加密策略。

总结一下, 本实验不仅帮助我们了解了不同平台在 HTTP 协议选择、页面加载优化及安全防护方面的实际应用, 同时也引发了对用户隐私保护的思考, 对今后网络流量分析和安全审计具有重要参考意义。

五.参考文献

[1]. <https://www.cnblogs.com/yurang/p/11505741.html>, Wireshark 解密 HTTPS 流量的两种方法, 豫让. 2022.3.15.

[2] https://blog.csdn.net/weixin_45910068/article/details/126290248, 详解 HTTP 协议版本 (HTTP/1.0、1.1、2.0、3.0 区别), -YIN, 2021.9.14

[3] <https://blog.csdn.net/zzwwhphp/article/details/113077747> 网络分析工具——WireShark 的使用 (超详细) .0x00dream. 2025.1.14

[4] <https://blog.csdn.net/walleva96/article/details/106844033>, 解密 TLS 协议全记录之利用 wireshark 解密, . walleVA96 . 2024.11.17

[5] <https://docs.pingcode.com/baike/2196612>: 前端如何分析网站, Edit1, 2023.6.6

[6] <https://www.qinglite.cn/doc/3539647668096e510>, 当你浏览网页时, 会被收集到哪些隐私?, 产品的技术小课, 2021.6.6