从英语翻译成中文(简体) - www.onlinedoctranslator.com

实验室检查点 5：堆栈（网络接口）

到期的：11 月 19 日（周五），下午 5 点 最迟
截止期限：11月21日晚上11点59分

0 合作政策

编程作业必须是你自己的作品：您必须编写您提交的编程作业的所有代码，除了我们作为作业的一部分提供给您的代码。请不要从 Stack Overflow、GitHub 或其他来源复制粘贴代码。如果您根据在 Web 或其他地方找到的示例编写自己的代码，请在您提交的源代码的注释中引用 URL。

与他人合作：您不得向其他人展示您的代码，不得查看其他人的代码，也不得查看前几年的解决方案。您可以与其他学生讨论作业，但不得抄袭任何人的代码。如果您与其他学生讨论作业，请在您提交的源代码的评论中注明他们的名字。请参阅课程管理讲义了解更多详细信息，如果有任何不清楚的地方，请在 Piazza 上询问。

广场：请随意在 Piazza 上提问，但请不要发布任何源代码。

1 概述

在本周的实验中，您将逐步实现网络接口：在世界各地传输的 Internet 数据报与传输一跳的链路层以太网帧之间的桥梁。此组件可以放在您之前实验中的 TCP/IP 实现的“下方”，但它也将用于不同的设置：当您在实验 6 中构建路由器时，它将路由数据报*之间*网络接口。图1显示网络接口如何适应这两种设置。

在过去的实验中，你编写了一个可以成功交换TCP 段 与其他任何使用 TCP 的计算机通信。这些段实际上是如何传达给对等方的 TCP 实现的？正如我们所讨论的，有以下几种选择：

- TCP 接入 UDP 接入 IP。TCP 段可以放在用户数据报的有效载荷中。在正常（用户空间）设置下工作时，这是最容易实现的：Linux 提供了一个接口（“数据报套接字”，UDP套接字允许应用程序提供*仅限有效载荷*用户数据报和目标地址的地址，内核负责构建 UDP 报头、IP 报头和以太网报头，然后将数据包发送到适当的下一跳。内核确保每个套接字都有唯一的本地和远程地址和端口号组合，而且由于内核将这些写入 UDP 和 IP 报头，因此它可以保证不同应用程序之间的隔离。

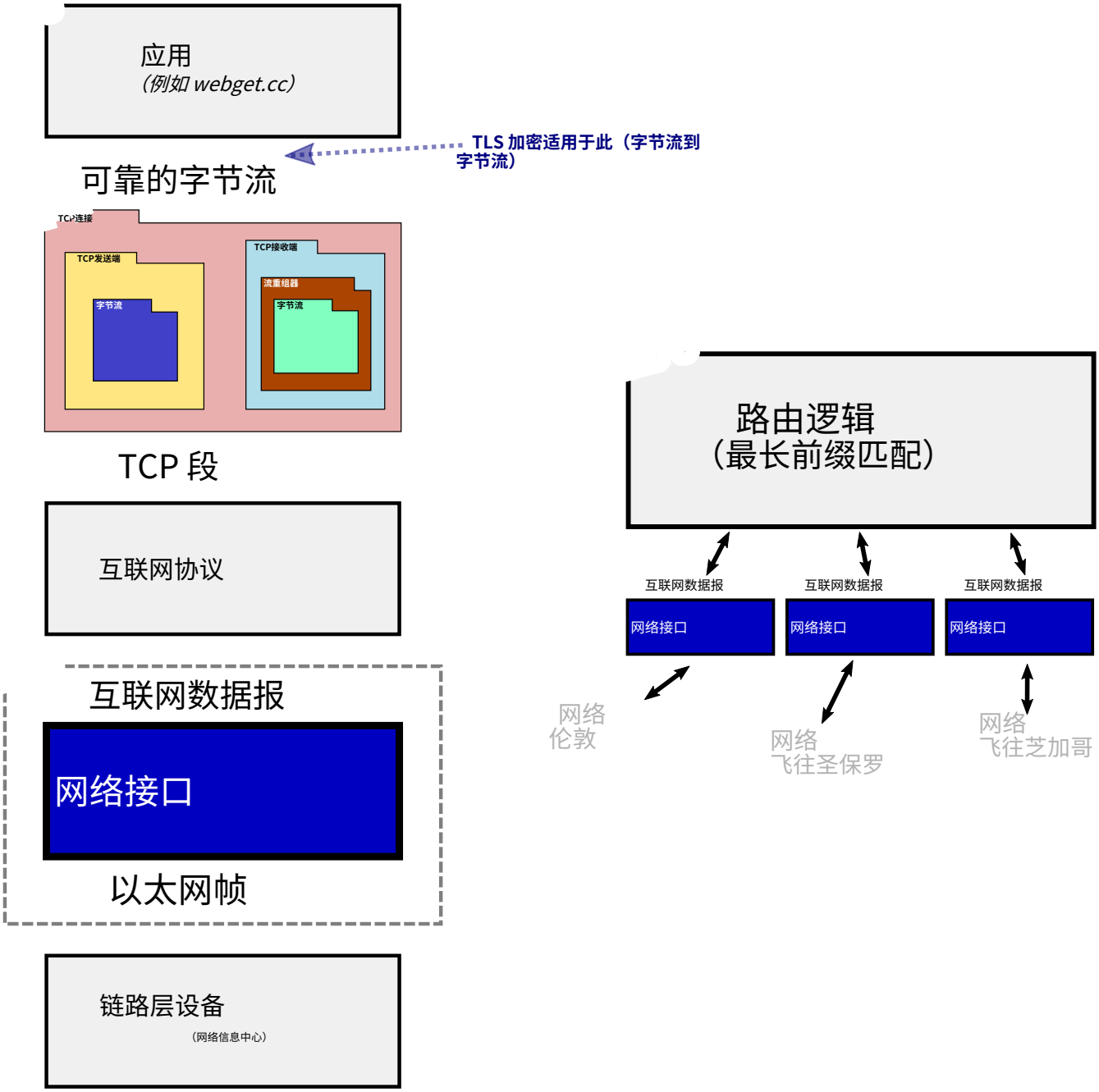

图 1：网络接口连接互联网数据报和链路层帧。此组件可用作主机 TCP/IP 堆栈的一部分（左侧），也可用作 IP 路由器的一部分（右侧）。

- TCP 和 IP 之间的连接。在常见用法中，TCP 段几乎总是直接放在 Internet 数据报中，IP 和 TCP 报头之间没有 UDP 报头。这就是人们所说的“TCP/IP”。这有点难以实现。Linux 提供了一个称为 TUN 设备的接口，它允许应用程序提供*全部的*互联网数据报，内核负责其余部分（编写以太网报头，并通过物理以太网卡实际发送等）。但现在应用程序必须自己构建完整的 IP 报头，而不仅仅是有效载荷。

  你已经完成了这个。在实验 4 中，我们给了你一个代表 Internet 数据报的对象，它知道如何解析和序列化自身（tcp 帮助程序/ipv4 数据报。*hh,抄送*）以及将 TCP 数据段封装在 IP 中的逻辑（现在位于tcp 帮助程序/tcp over ip.cc）。 这CS144TCP插座使用这些工具连接您的TCP连接到 TUN 设备。

- TCP 接入 IP 接入以太网。在上述方法中，我们仍然依赖 Linux 内核来实现部分网络堆栈。每次您的代码将 IP 数据报写入 TUN 设备时，Linux 都必须构建一个适当的链路层（以太网）帧，并将 IP 数据报作为其有效负载。这意味着 Linux 必须根据下一跳的 IP 地址找出下一跳的以太网目标地址。如果它还不知道此映射，Linux 会广播一个查询，询问“谁声明了以下 IP 地址？您的以太网地址是什么？”并等待响应。

  这些功能由*网络接口*：将出站 IP 数据报转换为链路层（例如以太网）帧或将链路层（例如以太网）帧转换为出站 IP 数据报的组件。（在实际系统中，网络接口通常具有如下名称eth0、eth1、wlan0、ETC。）在本周的实验室中，您将实现一个网络接口，并将其置于 TCP/IP 堆栈的最底层。您的代码将生成原始以太网帧，这些帧将通过称为 TAP 设备的接口移交给 Linux — 类似于 TUN 设备，但级别更低，因为它交换原始链路层帧而不是 IP 数据报。

大部分工作都是查找（并缓存）每个下一跳 IP 地址的以太网地址。用于此目的的协议称为地址解析协议，或者地址解析协议 (ARP)。

我们已为您提供单元测试，以测试您的网络接口的性能。然后，在本实验结束时，您将稍微修改您的网络获取使用你的 TCP 实现，*在您的网络接口上运行*，这样整个系统就会生成原始以太网帧，并且仍可以通过互联网与真实的网络服务器通信。在实验 6 中，您将在 TCP 上下文之外使用相同的网络接口作为 IP 路由器的一部分。

2 入门

1. 确保你已经提交了实验 4 的所有解决方案。请不要修改自由海绵目录，或webget.cc. (并且请不要添加您的代码所依赖的额外文件。）否则，您可能会在合并 Lab 5 启动代码时遇到麻烦。

2. 在实验室作业存储库中，运行实验室作业的最新版本。 git 获取 检索

3. 运行以下命令下载实验 5 的起始代码 git 合并 origin/lab5-startercode。

4. 在您 的建造目录，编译源代码： 制作 （例如，你可以运行 制作-j4 在编译时使用四个处理器）。

5. 外面建造目录，打开并开始编辑writeups/lab5.md fi这是您的实验室报告模板，将包含在您的提交内容中。

3 地址解析协议

在开始编码之前，请阅读：

- 这公共接口网络接口目的。
- 维基百科ARP 概述和原版ARP 规范（RFC 826）。
- 记录/实施以太网帧和以太网报头对象。
- 文件和实施IPv4数据报， 和IPv4标头对象（可以解析和序列化互联网数据报，并且在序列化后可以分配给以太网帧的有效负载）。
- 文件和实施ARP消息对象（它知道如何解析和序列化 ARP 消息，并且在序列化时也可以作为以太网帧的有效负载）。

您在本实验中的主要任务是实现以下三种主要方法网络接口 （在网络接口.cc file），维护从 IP 地址到以太网地址的映射。映射是一个缓存，或“软状态”：NetworkInterface 出于效率原因保留它，但如果必须从头重新启动，映射将自然地重新生成，而不会导致问题。

1.空白网络接口::发送数据报（常量互联网数据报&dgram，
常量地址&下一跳）

此方法在调用者（例如，您的TCP连接或路由器）想要将出站 Internet（IP）数据报发送到下一跳。[1]您的接口的工作是将该数据报转换为以太网帧并（最终）发送它。

- *如果目标以太网地址已知*，立即发送。创建一个以太网帧（使用类型 = EthernetHeader::TYPE IPv4)将有效载荷设置为序列化的数据报，并设置源地址和目标地址。

[1]请不要混淆*最终的*数据报的目的地，即数据报本身报头中的目的地地址，以及下一跳。在本实验中，您将*仅有的*去关心下一跳的地址。

- *如果目标以太网地址未知*，广播对下一跳的以太网地址的 ARP 请求，并将 IP 数据报排队，以便在收到 ARP 回复后发送。

除了：你不会希望 ARP 请求充斥整个网络。如果网络
如果接口在过去五秒内已经发送了有关同一 IP 地址的 ARP 请求，则不要发送第二个请求 — 只需等待对第一个请求的回复。再次对数据报进行排队，直到您了解目标以太网地址。

2.选修的<互联网数据报>网络接口::接收帧（常量以太网帧&框架）

当以太网帧从网络到达时，将调用此方法。代码应忽略任何不是发往网络接口的帧（这意味着以太网目的地是广播地址或存储在以太网地址成员变量）。

- *如果入站帧是 IPv4*，将有效载荷解析为互联网数据报如果成功的话（意味着解析（）方法返回ParseResult ::无错误） 返回结果互联网数据报给呼叫者。

- *如果入站帧是 ARP*，将有效载荷解析为ARP消息如果成功，则记住发送方的 IP 地址和以太网地址之间的映射 30 秒。（从请求和回复中学习映射。）此外，如果是询问我们 IP 地址的 ARP 请求，则发送适当的 ARP 回复。

3.空白网络接口::打钩（常量size_t自上次滴答以来的毫秒数）

这称为随着时间流逝。使所有已过期的 IP 到以太网映射失效。

您可以通过运行来测试您的实现 不依赖于您的 TCP `ctest-V-R “^arp”`。这个测试实现。

4 webget 重新2访问过

记住你的网络获取你在实验 0 中编写的代码（使用 Linux 提供的 TCP 实现TCPSocket 接口？还记得在实验 4 中你是如何修改它的，以便在CS144TCPSocket？正如我们上面讨论的，这仍然依赖于 Linux 内核的部分堆栈：在 IP 和链路层（以太网）之间转换的网络接口。

我们希望您将其切换到使用您的网络接口，而无需进行任何其他更改。您需要做的就是替换 `CS144TCP插座` 键入 `全栈套接字`。

这使用了 TCP-in-IP-in-Ethernet 堆栈，如图所示1（左侧）：你的网络获取 应用程序，在你的TCP连接TCP 的实现，在 TCP-in-IP 代码之上tcp 帮助程序/tcp over ip.cc，在你的网络接口。

重新编译并运行 检查 lab5 _ 确认你已经全栈了：你已经写了
一个基本的 Web 获取器*您自己的完整 TCP 实现*和*您自己的网络接口实现*，并且它仍然可以成功地与真正的网络服务器通信。

如果遇到问题，请尝试运行程序 man 实际上：
./apps/webget cs144.keithw.org/hasher/xyzzy ，并尝试捕捉它发送的内容
通过使用接收wireshark。您可以保存 pa 通过运行来发送和接收
sudo tcpdump -i tap10 -w /tmp/packets.tap 。然后打开 /tmp/packets.tap fi乐
在wireshark。

5 问答

- *您期望的代码量是多少?*

  总体而言，我们预计实施（在网络接口.cc)总共需要大约 100–150 行代码。

- *NetworkInterface 实际上如何发送以太网帧?*

  类似的故事TCP发送端和TCP连接有效。对于网络接口， 将任何出站帧推送到框架出来队列，它将被对象所有者弹出并发送。 _

- *我应该使用什么数据结构来记录下一跳 IP 地址和以太网地址之间的映射?*

  由你决定!

- *如何将以地址对象形式出现的 IP 地址转换为可以写入 ARP 消息的原始 32 位整数?*

  使用地址::ipv4 数字()方法。

- *如果 NetworkInterface 发送了 ARP 请求但从未收到回复，我该怎么办？我应该在超时后重新发送吗？使用 ICMP 向原始发送者发出错误信号?*

  在现实生活中，是的，这两件事都会发生，但在这个实验中不必担心。（在现实生活中，如果接口无法收到对其 ARP 请求的回复，它最终会通过 Internet 将 ICMP“主机不可达”发送回原始发送者。）

- *如果 InternetDatagram 排队等待了解下一跳的以太网地址，而该信息始终未到达，我该怎么办？我是否应该在超时后丢弃该数据报?*

  再说一遍，在现实生活中肯定是“是”，但在这个实验室里不用担心这个问题。

- *如何运行该实验室的测试套件?*

  检查 lab5 _ （两个测试）。或者你可以使用以下方法运行整个测试套件 进行检查
  （160 次测试）。

- *此 PDF 版本发布后，我可以在哪里阅读更多常见问题解答?*

  请查看网站（https://cs144.github.io/lab常见问题.html) 和 Piazza 经常光顾。

6 提交

1. 在您提交的作品中，请仅对 进行更改。时和 。cc fi位于顶层的libsponge，和应用程序/webget.cc。在这些文件中，请根据需要随意添加私有成员，但请不要更改*民众*任何类的接口。

2. 请不要添加额外的文件——自动评分器不会查看它们，并且您的代码可能无法编译。

3.请不要忘记改变你的网络获取使用全栈套接字 (而不是CS144TCP插座你在实验 4 中使用的，或者内核的TCP套接字在实验室 0 中)。

4. 在提交任何作业之前，请按顺序运行以下内容：

   （一）制作格式 （规范化编码风格）
   （二）git 状态 （检查未提交的更改 - 如果有，请提交！）
   （三）制作 （确保代码可以编译）
   （四）检查 lab5 （确保自动化测试通过）

5. 撰写报告writeups/lab5.md。该文件应为大约 20 到 50 行的文档，每行不超过 80 个字符，以方便阅读。报告应包含以下部分：

   （一个）程序结构和设计。描述代码中体现的高级结构和设计选择。您无需详细讨论从起始代码中继承的内容。利用这个机会突出重要的设计方面，并更详细地介绍这些方面，以便您的评分助教理解。强烈建议您使用小标题和大纲使本文尽可能易于阅读。请不要简单地将您的程序翻译成一段英文。

   （二）实施挑战。描述您发现最麻烦的代码部分并解释原因。反思您如何克服这些挑战以及是什么帮助您最终理解了让您感到困难的概念。您如何尝试确保您的代码保持您的假设、不变量和先决条件，以及在哪些方面您觉得这很容易或很难？您如何调试和测试您的代码?

   （三）餘下的错误。尽可能指出并解释代码中存在的任何错误（或未处理的边缘情况）。

6. 请填写完成该作业所花费的小时数以及其他评论。

7. 准备提交时，请按照以下说明操作https://cs144.github.io/提交。提交前请确保您已完成所有想要完成的工作。

8. 如果在周三晚上的实验课上遇到任何问题，请尽快告知课程工作人员，或者在 Piazza 上发布问题。祝您好运！