Tsinghua University
Department of Computer Science and Technology

Arrow4over6 隧道系统实验报告

瞿凡
2017010636

June 2020

Table of Contents

1. 内容介绍

1.1 实验目标

本实验的目标是实现一套 IPv4 over IPv6 的隧道系统，即 IPv6 的客户端可以通过 IPv6 网络与过渡网关建立连接，过渡网关将 IPv6 客户端的网络报文封装成 IPv4 报文与 IPv4 网络进行通信的过程。通信流程如 Figire 1.1 所示。

而本实验中实现的部分，是 IPv6 客户端的软件部分，具体为 Android 手机平台上的 App 软件，功能包括将 Android 上的网络流量通过虚拟网卡进行转发，通过 IPv6 Socket 与过渡网关建立连接，实现一套包含 keeplive 功能的隧道通信机制。过渡网关由实验指导者提供。该系统被我们命名为 Arrow4over6 隧道系统。

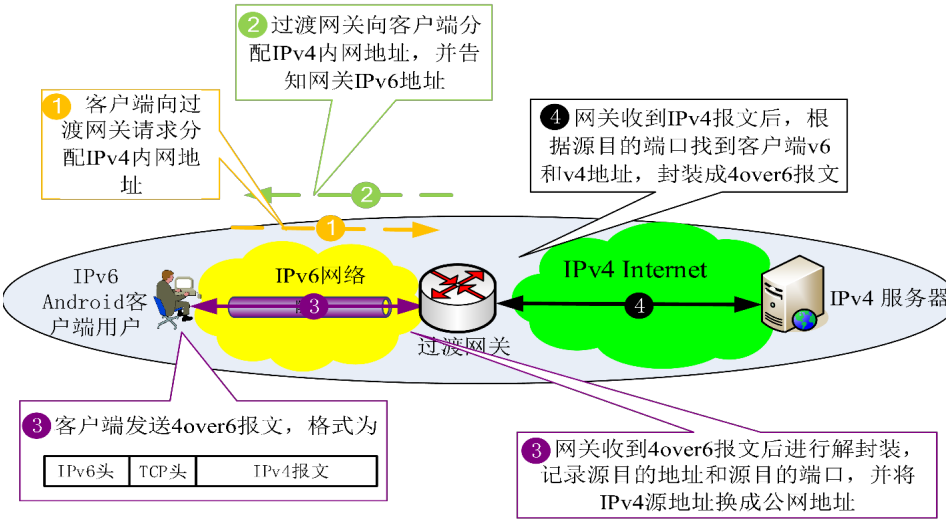

Figure 1.1: 4over6 隧道流程演示，来自 ppt

1.2 实验意义

通过该实验，我们理解并基本掌握了 IPv4 over IPv6 隧道的原理和实现，并且通过亲身实验掌握了在 Android 平台下开发代理通信软件的流程，对网络工程的认识得到了提升。

Arrow4over6 系统的使用情景，是在“先进”的只有 IPv6 网络的环境中，让 IPv6 终端可以通过与过渡网关建立隧道来访问 IPv4 虽然我们认为 IPv6 终网络的内容。将会是未来网络的主流，但是在发展的过程中总是有许多阻力与弯路，而我们也不过是体验了如何在发展过程中去克服一点阻力。至于这样的技术会不会发展出更多弯路，那又是另一个问题。

2. 实验原理

2.1 隧道原理

IPv4 over IPv6 隧道的原理并不复杂，IPv6 终端通过 IPv6 网络与过渡网关建立连接，过渡网关同时可以访问 IPv4 网络，IPv6 客户端将 IPv6 终端的一切对外网络报文封装成 IPv6 报文后传输给过渡网关，过渡网关拆解报文后通过 IPv4 的虚拟网卡发送出去。而接收报文的过程也是类似的。

通过这套机制，一个只有 IPv6 网络环境的终端可以访问 IPv4 网络的资源，当然这个原理也可以被用来建立 IPv6 over IPv4 隧道系统，或者说，这套系统可以建立起 $A\ over\ B, A, B \in \{ipv4, ipv6\}$ 的系统。当 $A = B$ 时，这个系统变为一个传统代理系统。

2.2 工作流程

这里简要介绍客户端的工作流程，客户端的流程分为前台线程组和后台线程组，前台负责处理交互与 UI、展示，后台处理网络连接与转发。在程序建立时即启动前台与后台线程。

前台线程流程如下

1. 等待后台建立 FIFO 管道并打开
2. 当连接按钮触发时，发送 IP 与端口到后台，code = 101
3. 读取后台发来的 IP 等信息，code = 201
4. 建立 VPN tunnel，将 vpnfd 发送给后台，code = 102
5. 循环读取管道中的信息，信息分为
   (a) code = 202，更新 UI 中的统计数据
   (b) code = 203，结束循环

后台线程组的流程如下

1. 建立有名 FIFO 管道并打开
2. 循环读取来来自管道中的信息，信息分为
   (a) code = 101, 读取到服务器 IP 和端口，建立 Socket 连接
      i. 向服务器请求 IP 等信息
      ii. 接受来自服务器的 IP 信息
      iii. 发送 sockfd,IP 信息到前台，code = 201
   (b) code = 102，读取前台建立的 vpnfd，开启 3 个子线程
      i. 循环从 vpnfd 虚接口中读取数据，写入 socket 中
      ii. 循环从 socket 中读取数据，写入 vpnfd 中
      iii. 定时线程，处理 keeplive 机制，并且将统计的流量信息发送至前台，code = 202
   (c) code =103, 结束连接，断开 socket

3. 工程架构

3.1 构成模块

Arrow4over6 的客户端主体分为前台与后台，前台包括 UI 系统、交互处理、数据展示等，后台包括网络连接、数据转发、数据统计等功能。

在实现上，前台由 Java 实现，后台由 C++ 实现，在程序开始时前台启动后台子线程，前台后台的一切数据交互都通过具名 FIFO 管道，具体流程见2.2。

前台的各个类设计如 Figure3.1所示。

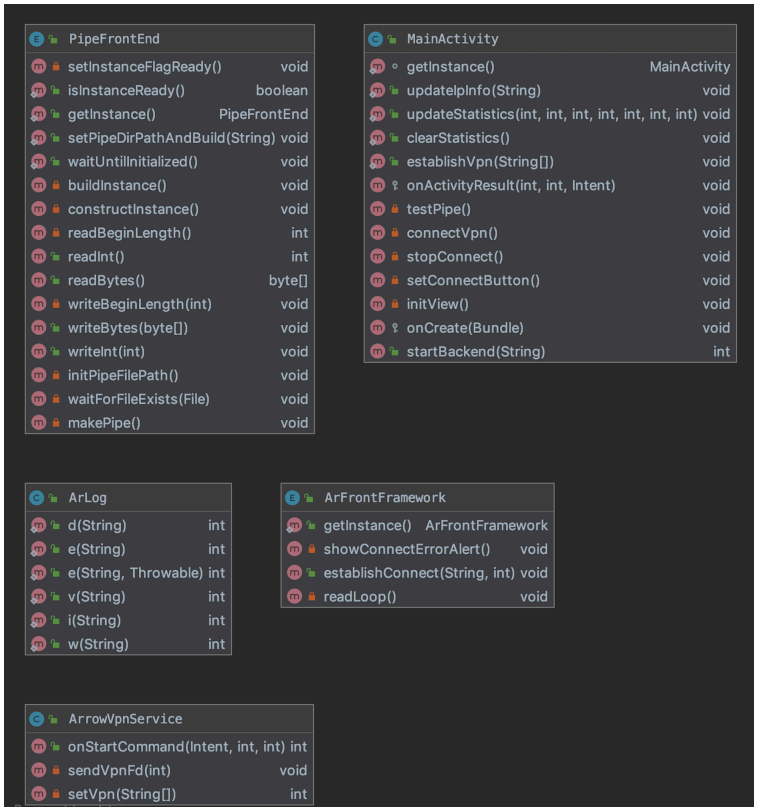

Figure 3.1: Frontend UML Diagram

前台模块构成如 Table 3.1所示。

文件	功能
ArFrontFramework.java	前台连接管理框架
ArrowVpnService.java	VpnService 服务
PipeFrontEnd.java	FIFO 管道前端单例
MainActivity.java	MainActivity 主 UI
Constants.java	常数定义
ArLog.java	Log wrapper

Table 3.1: 前台程序文件结构

后台程序组织如 Table 3.2所示。

文件	功能
Singleton.h	单例模式模板类
ArBackFramework.h	后台主框架类
PipeBackend.h & PipeBackend.cpp	FIFO 管道后端单例
SocketPlugin.h & SocketPlugin.cpp	Socket 管理单例
arrowutils.h & arrowutils.cpp	简单工具
arrowlib.cpp	JNI 程序入口

Table 3.2: 后台程序文件结构

3.2 设计原因

分成前后台设计的原因，包括这样分离设计可以将不同的任务交由更适合的工具进行。Java 与 Android 负责交互与 UI，C++ 与 Linux API 负责网络栈部分。通过 FIFO 管道连接的效率对于这样交互不算十分频繁的程序来说是完全足够的。

4. 实验结果

4.1 运行示范

在建立连接前，Android App 的 UI 如 Figure 4.1所示，建立连接后的 UI 如 Figure 4.2 所示。

Figure 4.1: 未连接时的 UI

在隧道建立后，由于过渡网关在清华校园内，因此可以访问清华内网，访问 info 网站的截图如 Figure 4.3所示。

Figure 4.2: 连接时的 UI

Figure 4.3: 访问 info 时的截图

5. 实验结论

我们在这次试验中实现的 Arrow4over6 系统是对 Ipv4 over Ipv6 隧道的一个实现。通过这次实验，我们验证了该机制的可行性，并且加深了对于 IPv6 网络的理解。根据2.1中所讨论，其实这次实验不限于 Ipv4 over IPv6，这个机制的本质是一个代理隧道，任何网络都可以组成。只不过 IPv4 over IPv6 有自己的独特现实意义。

最后，感谢老师和助教的帮助。