云南大学数学与统计学院
《计算机网络实验》上机实践报告

课程名称： 计算机网络实验	**年级：** 2015 级	**上机实践成绩：**
指导教师： 陆正福	**姓名：** 刘鹏	**专业：** 信息与计算科学
上机实践名称： 候选平台预备实验	**学号：** 20151910042	**上机实践日期：** 2018-09-01
上机实践编号： No.02	**组号：**	

一、 实验目的

1. 熟悉本学期候选编程平台 Python，为后继实验奠定候选基础。
2. 熟悉本学期候选编程平台 Android，为后继实验奠定候选基础。
3. 熟悉本学期候选协议分析软件 Wireshark，为后继实验奠定候选基础。

二、 实验内容

1. 查阅 Python 官方网站，下载最新版本软件，安装、配置并测试。编制调试典型的示例程序。
2. 查阅 Android 开发的官方网站，下载最新版本软件，安装、配置并测试。编制调试典型的示例程序
3. 查阅 Wireshark 官方网站，下载最新版本软件，安装、配置并测试。

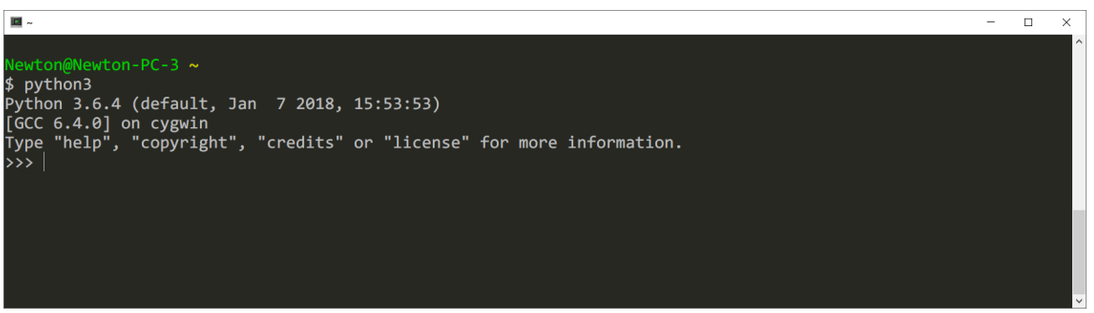
三、 实验平台

Windows 10 Pro Workstation 1803；
Cygwin GCC 编译器。

四、 算法设计

五、 程序代码

1.1 Python 环境的搭建



本次实验采用 Cygwin 平台的 Python3 进行实验，目前来看，除了 matplotlib 无法简单地进行配置使用外，其他方面与 Windows 版的 Python3 没有区别。

1.2 Android 开发环境的安装配置

Android 系统是由 Andy Rubin 创建的，后来被 Google 收购；最早的版本是 Android 1.1，现在已经更新到了 Android 9.0。Android 的开发套件由谷歌公司提供，在 Windows 平台下有 Android Studio 这款 IDE 可以使

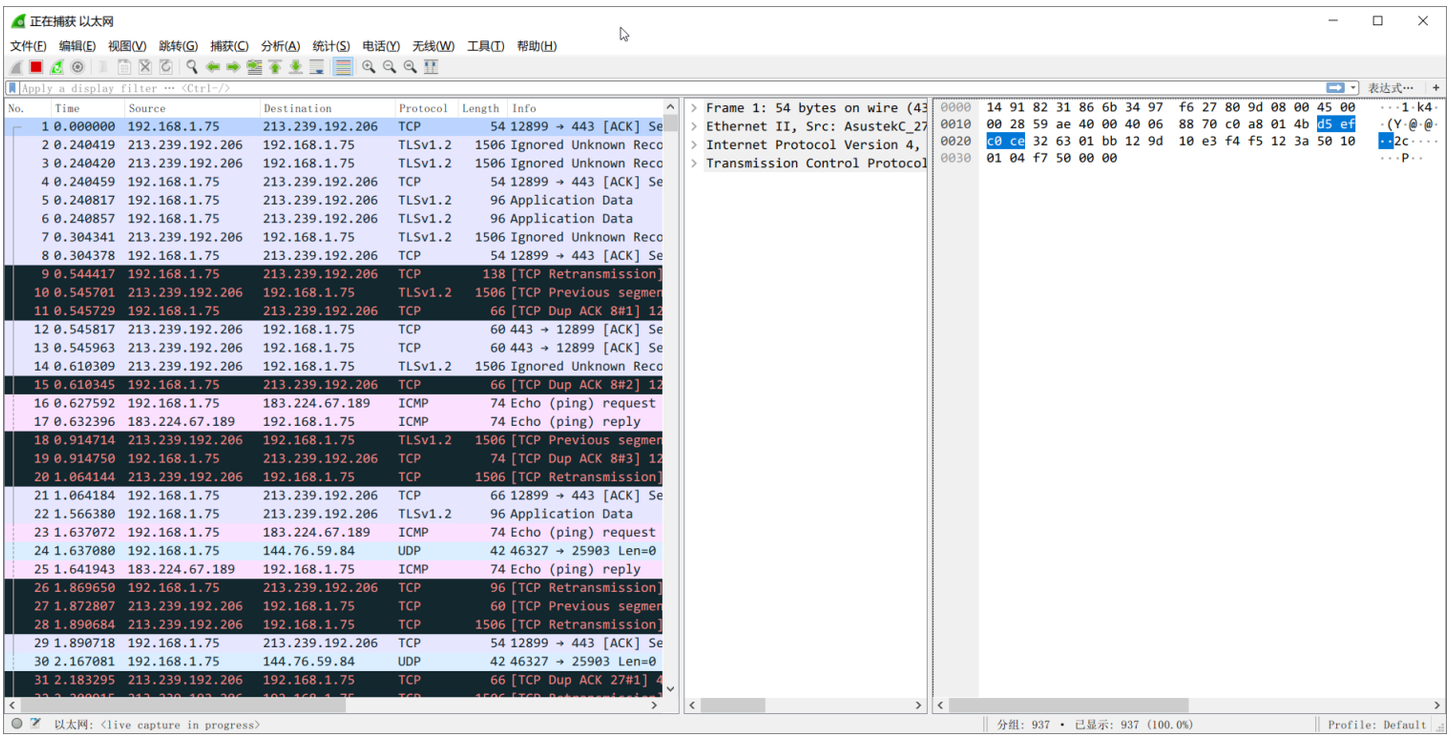
用。Android 的开发语言是 Java，配置了 Android SDK 与 Java SDK 之后，就可以进行 Android 开发。（不太懂为什么要用安卓进行开发测试）

1.3 Wireshark 抓包软件的安装配置

Wireshark 是一个网络 packet 嗅探软件，该软件可以跨平台。Wireshark 可以运行在命令行中，在 UNIX 下，如 MacOS，可以通过命令行界面进行调用，但是在 Windows 下比较繁琐，由于 Windows 的命令行十分难用，这里就不在 CLI 下进行实验了。

下面展示的是通过 Git 进行 ssh 的 Repo clone 时，wireshark 所捕获的一些信息。由于 Git 采用的是 ssh 协议，所以在 Protocol 项里显示的应该是 ssh。

如下图，由于网卡这个硬件被 Windows 10 系统互斥地使用，所以在一个任务的执行期间，网卡也可以收发其他 packet。



在对 Protocol 进行排序之后，可以看到每个 ssh 协议下的 packet 之间，时间间隔非常小，每个项目的编号 No 也是基本连续。而且通过 Info 项可以看到，整个过程，起于本 host（即 client）进行呼叫，然后 GitHub 的服务器（Server）进行应答，并且 Server 会发起一个密钥交换的初始化过程。通过椭圆曲线类型的 Diffie-Hellman 密钥交换过程之后（#22 与#26），Client 与 Server 都获得了一个新的密钥 New Keys，然后就开始了大规模的 packet 传输。

在此过程中，可以看到#93 发生了错误。最终在#137 完成之后，整个 git clone 命令执行完毕。可以看到，在 git clone 过程里，git 调用 clone 命令，以 URL 为参数，向 GitHub 发起通讯，建立连接之后，完成密钥交换并以次密钥为加密、解密的基础，进行加密 packet 的传输。此外，GitHub 还支持 HTTPS 协议，这时实验发现该过程并没有一个名为 HTTPS 的 Protocol 项出现，而是一种名为 TLSv1.2 的协议。

No.	Time	Source	Destination	Protocol	Length	Info
8	0.183781	192.168.1.75	13.229.188.59	SSHv2	75	Client: Protocol (SSH-2.0-OpenSSH_7.7)
19	0.734037	13.229.188.59	192.168.1.75	SSHv2	604	Server: Protocol (SSH-2.0-libssh_0.7.0), Key Exchange Init
20	0.734698	192.168.1.75	13.229.188.59	SSHv2	1350	Client: Key Exchange Init
22	0.807557	192.168.1.75	13.229.188.59	SSHv2	102	Client: Elliptic Curve Diffie-Hellman Key Exchange Init
26	1.129670	13.229.188.59	192.168.1.75	SSHv2	662	Server: Elliptic Curve Diffie-Hellman Key Exchange Reply
27	1.129670	13.229.188.59	192.168.1.75	SSHv2	70	Server: New Keys
29	1.131748	192.168.1.75	13.229.188.59	SSHv2	70	Client: New Keys
31	1.203958	192.168.1.75	13.229.188.59	SSHv2	118	Client: Encrypted packet (len=64)
40	1.821029	13.229.188.59	192.168.1.75	SSHv2	118	Server: Encrypted packet (len=64)
41	1.821241	192.168.1.75	13.229.188.59	SSHv2	134	Client: Encrypted packet (len=80)
45	2.137655	13.229.188.59	192.168.1.75	SSHv2	118	Server: Encrypted packet (len=64)
46	2.137886	192.168.1.75	13.229.188.59	SSHv2	438	Client: Encrypted packet (len=384)
48	2.459315	13.229.188.59	192.168.1.75	SSHv2	390	Server: Encrypted packet (len=336)
49	2.460678	192.168.1.75	13.229.188.59	SSHv2	710	Client: Encrypted packet (len=656)
51	2.777366	13.229.188.59	192.168.1.75	SSHv2	102	Server: Encrypted packet (len=48)
52	2.777648	192.168.1.75	13.229.188.59	SSHv2	134	Client: Encrypted packet (len=80)
58	3.094120	13.229.188.59	192.168.1.75	SSHv2	118	Server: Encrypted packet (len=64)
59	3.096331	192.168.1.75	13.229.188.59	SSHv2	166	Client: Encrypted packet (len=112)
61	3.412517	13.229.188.59	192.168.1.75	SSHv2	102	Server: Encrypted packet (len=48)
62	3.441407	13.229.188.59	192.168.1.75	SSHv2	374	Server: Encrypted packet (len=320)
64	3.441885	13.229.188.59	192.168.1.75	SSHv2	182	Server: Encrypted packet (len=128)
65	3.441930	13.229.188.59	192.168.1.75	SSHv2	166	Server: Encrypted packet (len=112)
66	3.441930	13.229.188.59	192.168.1.75	SSHv2	118	Server: Encrypted packet (len=64)
68	3.452561	192.168.1.75	13.229.188.59	SSHv2	374	Client: Encrypted packet (len=320)
70	3.768962	13.229.188.59	192.168.1.75	SSHv2	118	Server: Encrypted packet (len=64)
71	3.769498	13.229.188.59	192.168.1.75	SSHv2	118	Server: Encrypted packet (len=64)
73	3.776450	13.229.188.59	192.168.1.75	SSHv2	118	Server: Encrypted packet (len=64)
74	3.776451	13.229.188.59	192.168.1.75	SSHv2	134	Server: Encrypted packet (len=80)
76	3.777400	13.229.188.59	192.168.1.75	SSHv2	118	Server: Encrypted packet (len=64)
77	3.777502	13.229.188.59	192.168.1.75	SSHv2	182	Server: Encrypted packet (len=128)
79	3.778007	13.229.188.59	192.168.1.75	SSHv2	118	Server: Encrypted packet (len=64)
80	3.778134	13.229.188.59	192.168.1.75	SSHv2	150	Server: Encrypted packet (len=96)
81	3.778135	13.229.188.59	192.168.1.75	SSHv2	118	Server: Encrypted packet (len=64)
83	3.778329	13.229.188.59	192.168.1.75	SSHv2	150	Server: Encrypted packet (len=96)
87	4.012765	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
88	4.012766	13.229.188.59	192.168.1.75	SSHv2	1466	Server: Encrypted packet (len=1412)
90	4.013293	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
91	4.013293	13.229.188.59	192.168.1.75	SSHv2	1466	Server: Encrypted packet (len=1412)
93	4.020205	13.229.188.59	192.168.1.75	SSHv2	1466	Server: [TCP Previous segment not captured] , Encrypted packet (len=1412)
95	4.020207	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
98	4.020285	13.229.188.59	192.168.1.75	SSHv2	1466	Server: Encrypted packet (len=1412)
100	4.021209	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
101	4.021209	13.229.188.59	192.168.1.75	SSHv2	1466	Server: Encrypted packet (len=1412)
103	4.021433	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
104	4.021434	13.229.188.59	192.168.1.75	SSHv2	1466	Server: Encrypted packet (len=1412)
106	4.021935	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
107	4.021936	13.229.188.59	192.168.1.75	SSHv2	1466	Server: Encrypted packet (len=1412)
109	4.022083	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
110	4.022083	13.229.188.59	192.168.1.75	SSHv2	1466	Server: Encrypted packet (len=1412)
111	4.022084	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
112	4.022084	13.229.188.59	192.168.1.75	SSHv2	1466	Server: Encrypted packet (len=1412)
114	4.022156	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
115	4.022157	13.229.188.59	192.168.1.75	SSHv2	1466	Server: Encrypted packet (len=1412)
117	4.256573	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
118	4.256574	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
119	4.256575	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
120	4.256575	13.229.188.59	192.168.1.75	SSHv2	1442	Server: Encrypted packet (len=1388)
122	4.257007	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
123	4.257008	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
124	4.257008	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
125	4.257008	13.229.188.59	192.168.1.75	SSHv2	1442	Server: Encrypted packet (len=1388)
127	4.263873	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
128	4.263874	13.229.188.59	192.168.1.75	SSHv2	1490	Server: Encrypted packet (len=1436)
129	4.263875	13.229.188.59	192.168.1.75	SSHv2	974	Server: Encrypted packet (len=920)
131	4.287896	192.168.1.75	13.229.188.59	SSHv2	102	Client: Encrypted packet (len=48)
133	4.604257	13.229.188.59	192.168.1.75	SSHv2	134	Server: Encrypted packet (len=80)
134	4.604291	13.229.188.59	192.168.1.75	SSHv2	150	Server: Encrypted packet (len=96)
136	4.604450	192.168.1.75	13.229.188.59	SSHv2	102	Client: Encrypted packet (len=48)
137	4.604484	192.168.1.75	13.229.188.59	SSHv2	134	Client: Encrypted packet (len=80)

六、 实验体会

Python3 开发已经经历了相当长的过程，现在搭建环境相对比较熟悉；Android 环境搭建相对也比较简单，但是没有 Java 的开发基础，对于工程性的开发还有待进一步的学习，在实验过程中并没有给出很有代表性的例子，对于 Android 开发可能带来的网络编程收获也不甚了解；Wireshark 软件给出的信息，可以加深对于网络协议的理解，非常生动形象。

七、 参考文献

[1] **林锐**. 高质量 C++/C 编程指南 [M]. 1.0 ed., 2001.

[2] Android Development: https://developer.android.com/

[3] Wireshark: https://www.wireshark.org/

[4] ECKEL B. Java 编程思想 [M]. 4th ed. 北京: 机械工业出版社, 2007.