

南京邮电大学

毕 业 设 计（论 文）

题 目	基于 Linux 操作系统文件加密传输技术
专 业	网络工程
学生姓名	吴海东
班级学号	B13070221
指导教师	孙知信
指导单位	物联网学院

日期：2017 年 1 月 15 日至 2017 年 6 月 17 日

毕业设计（论文）原创性声明

本人郑重声明：所提交的毕业设计（论文），是本人在导师指导下，独立进行研究工作所取得的成果。除文中已注明引用的内容外，本毕业设计（论文）不包含任何其他个人或集体已经发表或撰写过的作品成果。对本研究做出过重要贡献的个人和集体，均已在文中以明确方式标明并表示了谢意。

论文作者签名：

日期： 年 月 日

摘 要

随着互联网的发展，网络环境变得越来越复杂。每天都有大量的总要文件通过网络进行传输，从日常人类社交，到企业文件传输，到军方机密信息输送，文件传输的安全性变得尤为重要。大量嵌入式应用的系统运行平台已经从私有操作系统、塞班系统、Windows CE 等过渡到开放性的 Linux 平台。大量的制造企业也已经在使用 Linux 平台进行代码研发。代码成果何其宝贵，如果有丝毫的泄露，都会造成巨大的损失。银行，军方等重要机构也使用 Linux 操作系统来实现文件的存储与传输。

本文探究传统的网络文件传输协议 SSL 协议，SSL 是一个提供 Internet 上的通信隐私性的安全协议，该协议允许客户端与服务器应用之间进行防窃听、消息篡改及消息伪造的安全通讯。这种协议有点很多，缺点也很明显，缺乏安全性，为此本文提出了 RSA 加密算法，RSA 算法的核心是实现了非对称加密，每次加密过程都用到公钥和私钥，保证了传输过程中密钥的安全性和文件的私密性，保障服务器安全和密钥的私密性。

本文的实验结果表明数据在 RSA 加密算法的处理下，可以实现文件的传输。在现有成熟的网络传输协议的基础上，通过 RSA 加密算法对数据进一步加密处理，具备实际应用价值。

关键词：Linux 操作系统；安全；文件传输；加密

ABSTRACT

With the development of the Internet, the network environment becomes more and more complicated. Every day there is a large number of documents to be transmitted through the network, from daily human social, to corporate document transmission, to the military confidential information transmission, file transmission security has become particularly important. A large number of embedded applications system operating platform has been from the private operating system, Saipan system, Windows CE and other transition to open Linux platform. A large number of manufacturing enterprises have also been using the Linux platform for code development. How the results of the code valuable, if there is the slightest leak, will cause a huge loss. Banks, military and other important institutions also use the Linux operating system to achieve the file storage and transmission.

This article explores the traditional network file transfer protocol SSL protocol, which is a security protocol that provides privacy on the Internet. This protocol allows secure communication between client and server applications for anti-eavesdropping, message tampering, and message forgery. This algorithm is a bit a lot, the shortcomings are obvious, the lack of security, this paper proposed RSA encryption algorithm, RSA algorithm is the core of the realization of asymmetric encryption, each encryption process are used public key and private key, to ensure that The security of the key during transmission and the privacy of the file, to ensure the security of the server and the privacy of the key.

In this paper, the experimental results table name data in the RSA encryption algorithm, you can achieve the file transfer. On the basis of the existing mature network transmission protocol, the RSA encryption algorithm is used to encrypt the data further, and it has practical application value.

Keywords: Linux operating system; security; file transfer; encryption

第一章 绪言.....	6 -
1.1 课题来源及背景.....	6 -
1.2 研究加密传输的意义.....	6 -
1.3 本文主要研究内容.....	8 -
第二章 SSL 协议以及 Linux 系统加密算法相关研究.....	9 -
2.1Linux 系统.....	9 -
2.1.1 系统概述.....	9 -
2.1.2 系统环境安全漏洞.....	9 -
2.2SSL 协议.....	10 -
2.2.1 SSL 协议组成.....	11 -
2.2.2 SSL 协议的应用.....	12 -
2.2.3 OpenSSL 简介.....	12 -
2.2.4 SSL 协议在 OpenSSL 中的的实现.....	12 -
2.2.5 SSL 协议优缺点分析.....	16 -
2.3 文件传输加密算法基础.....	16 -
2.3.1 对称加密算法.....	16 -
2.3.2 消息摘要加密算法.....	17 -
2.4 本章小结.....	18 -
第三章 一种改进的 RSA 加密算法.....	19 -
3.1 改进的 RSA 加密算法.....	19 -
3.2 加密流程.....	20 -
3.2.1 服务端加密流程设计.....	20 -
3.2.2 客户端加密流程设计.....	22 -
3.2.3 服务器端与客户端交互过程.....	23 -
3.2.4 传输及交互过程（一对一）.....	24 -
3.2.5 引入线程池（多对多）.....	24 -
3.3RSA 算法分析.....	25 -
3.4 本章小结.....	25 -
第四章 非对称加密算法的实验结果.....	26 -
4.1 登录验证与连接测试.....	26 -
4.2 上传测试.....	26 -
4.3 下载.....	27 -
4.4 本章小结.....	27 -
第五章 总结.....	28 -
结束语.....	29 -
致 谢.....	30 -
参考文献.....	31 -

第一章 绪言

1.1 课题来源及背景

互联网在世界的发展变化中起到了巨大的推动作用，给社会发展创造了是硕果累累，但同时也为人类带来了许多担忧与隐患，在如今信息泛滥，社会中信息的安全性占据举足轻重的位置，每天都有大量的重要文件通过网络进行传输，无论是人们日常社交还是企业或者是军方的信息传递，对数据传递的稳定性有很高的要求，而其中数据安全面临更大的威胁与挑战，网络社会所受到的威胁与侵害很大程度上是人为的甚至是无意的非法或合法用户的攻击。现在的开源软件所占比重比较小，而 Linux 系统恰巧是这些开源软件中为数不多的系统软件之一。由于其开源性及自由性，加之具有稳定、安全、网络负载力强、占用硬件资源少这些适用于系统开发与使用的优点，该系统已经迅速地得到了人们的认可与使用，在企业及家用的众多操作系统中占据了无法取代的位置。本文在 Linux 环境下对文件安全传输进行了研究，实现了数据的安全性传输。

文章在 Linux 环境下，对文件加密传输技术进行了研究，通过加密算法，对文件数据的加密复杂化，是的文件数据在服务器之间的传输，更加稳定，也更加安全。并且利用线程池的方法，实现多个客户端与服务端之间文件传输的稳定与数据文件的保密性。

1.2 研究加密传输的意义

互联网技术与计算机技术的高速发展，推动了网络安全（Network security system）的诞生和发展^[1]。网络安全是一种网络系统，它可以保护网络不被非法攻击，保障用户的信息安全，以保护网络信息的完整性，可用性，保密性，真实可靠性为目的的各种技术与方案的总和^{[2][3]}。他由各种网络层的软硬件构成^{[4][5][6]}，通过软件硬件、一级各种网络层的相互合作来采集、抓获、加密解密、压缩、处理网络传输过程中的各种数据对象所包含的信息^[7]。随着互联网的发展，它作为成长起来的新型研究领域，它对互联网的发展依赖而又影响着。从某一方面来说，没有互联网的诞生与发展，就没有网络安全的产生和发展，从另一方面来说，没有网络安全的发展，互联网也无法维持正常运行，也无法保证依赖互联网的人民的日常生活和社交活动。互联网应用的巨大潜力导致了网络安全诞生与飞速发展。网络安全已成为当今世界的研究热点，备受各国的关注。FTP（文件传输协议）系统的安全也属于网络安全领域中的一个^{[1][2][3][8]}。

在过去，FTP 文件传输服务在传输、存放大量的技术文档、共享软件^{[4][6][9]}这一领域有着巨大贡献，随着互联网的发展，FTP 文件传输服务在各种领域^{[4][9]}依然

有着巨大应用,而且它所应用的面越来越广泛,FTP 文件传输服务甚至于将视频^[11]、图像^[12]、音频^[10]等多媒体信息都囊括进来,为了给用户提供全方位的服务。大部分文件传输系统只想到了简单的信息传输,很少一部分考虑到其安全性,随着用户的增加、互联网环境的复杂化,过去的 FTP 文件传输系统已经无法满足当前互联网环境的更高需求,急切需要将安全领域有关的研究与技术引用到文件传输系统中,以此保障文件传输系统的安全。由此,安全文件传输系统(Security File Transfer Sysstem)应运而生^{[13][14][15]}。

安全文件传输系统不仅仅关注文件的传输,更多的是关注数据信息传送的安全,它是一种新型的文件传输系统。文件传输系统文件传输协议(File Transfer Protecol,FTP)与传输层两个端口之前加了一个隧道,保证了文件传输的同时,还能保障数据的安全,简单点来说就是在文件传输协议与传输层之间加了一个保护(加密)。文件数据的安全有两种情况,一种是在传输过程中就被不法分子截取,还有一种是即使被得到,也无法读取其中的数据,即对数据经过各种处理(如加密),非法者没有密钥无法解读其中的数据。过去一般使用安全外壳(Secure Shell,SSH)来保障 FTP 的安全性^{[16][17][18]},而安全套接字(Secure Socket Layer,SSL)大部分情况是用来加强 WEB 的安全。SSL 的优点十分突出显著,使得 SSL 在各种领域的应用迅速扩大,不仅仅在 FTP 上。SSL 在 FTP 上的应用,就是在应用层与传输层之间加上一层保护协议,从而对传输数据进行安全处理,从而保证了 FTP 数据的完整性、保密性、可控性、可靠性等^{[19][20]}。SSL 协议可以调动、协调各种加密算法,保密数据,它不仅能协同各种对称、不对成加密算法之间的合作,还能协调网络层各层协议集中合作处理数据(协调 TCP/IP 等协议之间的合作),将数据加密后传输到目标主机上或者远程服务器上,然后由目标主机或者服务器对数据进行反向分析,通过密钥解密,从而实现文件安全传输的目的。

传统的文件传输是明文传输,黑客可以通过暴力破解,获取用户的账号与密码,简单点来说就是盗号(如钓鱼网站),获得访问系统的权限,远程登陆盗取信息,没有任何安全保障。

还有一种是 DOS/DDOS 攻击,拒绝服务攻击与分布式拒绝服务攻击。它是一种破坏式的攻击,它的目的不是为了获取你的账户密码,而是利用这种攻击手段使你的服务器瘫痪(拒绝正常服务),无法正常访问,一般多见于企业服务器。

当然还有许多别的攻击,所以文件传输必须得到重视。

IPSec(Internet Protocol Serurity)是一种传统文件传输模型,它采用加密传输,但是它在网络层甚至更底层加密传输,不会对应用层数据进行加密,所以缺点也非常明显,一个是网络层或更底层的加密策略耗费成本,另一个缺点是应用层的攻击无法有效阻止。

我们通过 RSA 加密算法和 Linux 系统线程池技术的研究,实现服务器(基于 Linux 系统)与客户端的文件加密传输:

1) 在 Linux 系统上配置安装 OpenSSL 库,实现 RSA 非对称加密过程

2) 利用线程池技术实现一个服务器与多个客户端的文件传输，实现客户端服务器端的网络连接，并在 TCP/IP 协议上实现数据加密，保障了文件数据传输的安全性

结论表明在 SSL 协议上设计的加密算法能够完成加密，保障文件传输的安全，保障资料的私密性，具备实际的应用价值。

1.3 本文主要研究内容

本文首先讲述了越来越多嵌入式应用的系统在 Linux 平台运行。越来越多的互联网企业使用 Linux 平台研发代码。而 Linux 系统存在许多的漏洞，基于 Linux 系统的文件传输便是其中一个，在这里我们就需要引入 SSL 协议。

SSL 协议对于文件的安全传输有着必不可少的作用，它操作简单，对现有网络系统的修改少，而且速度快，成本低，因而在世界范围内饱受好评。但是 SSL 协议由于过于简单，容易被黑客破解，因而对基于 Linux 的文件传输继续探究，引入了加密算法。本文简单的对称加密算法和消息摘要加密算法进行了介绍。

接着本文提出了相对于上两种加密算法改进的 RSA 非对称加密算法，利用线程池多任务处理技术，利用网络传输协议实现在 Linux 平台文件的加密传输过程。其中 RSA 非对称加密算法系基于世界数学难题提出的加密算法且私钥独立持有，使之难以被黑客破解。

紧接着第四章是对非对称加密算法的实验认证，嵌入式技术的发展和 Linux 系统网络传输技术的普及会产生更多的网络安全问题，在现有成熟的网络传输协议的基础上，通过增加加密管道对数据进一步加密处理，能将加密过程很方便的移植到级别需求高的嵌入式系统中，具备实际应用价值。

最后本文对 Linux 文件加密传输做了总结。

第二章 SSL 协议以及 Linux 系统加密算法相关研究

第一章是对课题的来源与背景的研究，接下来这一章概述了 Linux 系统，对 SSL 协议做了详细的介绍，最后对传统加密算法的相关研究做了一些阐述

2.1Linux 系统

2.1.1 系统概述

从智能手机到汽车，超级计算机和家用电器，Linux 操作系统无处不在。

Linux 操作系统。自从 90 年代中期以来一直存在，并已经达到跨越工业和大陆的用户群。对于那些知道的人，你明白，Linux 实际上是无处不在的，在你的手机，汽车，冰箱，Roku 设备中。它运行大部分的互联网，超级计算机取得科学突破，以及世界的证券交易所。但是，在 Linux 成为全球运行台式机，服务器和嵌入式系统的平台之前，它是最可靠，安全，无忧的操作系统之一。

Linux 是一种类 Unix 操作系统^[21]，它是以网络为核心所设计的，是一个性能非常稳定的且支持多用户登录操作的网络操作系统。

严格来讲，Linux 只表示 Linux 内核，但人们习惯了用 Linux 来形容这种类 Unix 的操作系统。它整个基于 Linux 内核，并且使用 GNU 工程各种工具和数据库。

20 世纪 90 年代初，GNU 便开始集成各种功能模块，如代码编译器、各种功能的函数库、网页服务器、文档编辑器、纠错工具以及用户接口(一般叫 shell 程序)等。后来 GNU 在全世界各个国家的黑客的协同努力下，大力发展，在服务器、云计算等各方面迅速发展。为我们所熟知的著名的 Ubuntu (Linux 的一种版本)的发展有取代 windows 的可能，它受欢迎度将是 windows 未来重要的挑战。

2.1.2 系统环境安全漏洞

1、Linux 系统安全漏洞种类：

1)可以随意在 Linux 内核中增加各种模块。增加内核模块的两种方法：

一种是手动。通过 Linux 指令 insmod (install module) 载入模块。在需要时，insmod 命令可以在内核中手动加入各种内核模块。

第二种方法是自动装入。modprobe 可载入指定的任何模块或相关的模块。根据 depmod 产生的关系，modprobe 可以决定载入什么样的模块。这是 Linux 系统的优点，但从某方面来说，也恰恰可能成为了 Linux 系统安全所面临的最大的威胁。

2)缓冲区溢出漏洞，通过缓冲区溢出漏洞把破坏性代码写入指定内存空间，控制目标。

3)系统文档的安全漏洞，可以被轻易修改。Linux 系统中有非常重要的文件，黑客一旦黑入 Linux 系统便可随意修改文件，要么导致机密文件损坏无法读取，或

者导致系统直接瘫痪。

4) Linux 进程不受保护。

5) 系统管理员权限无限高，这种威胁主要是来自管理员本身，需要加强对管理员的监控与管理。

随着 Linux 用户的日益增加，Linux 还面临着其它各种各样的威胁。

2、应对以上几种漏洞，主要采取如下应对措施：

1) 对于内核模块可以被轻易加入的漏洞，我们主要采取 LSM，即对访问对象进行仲裁、控制、管制等措施，加以防范系统的恶意操作。

2) 对于缓冲区溢出威胁问题，可以使用基于探测方法防御或非执行堆栈防御等策略。

3) 对于文档可以轻易的被修改，防止主机被黑客或非授权用户控制而瘫痪主或者增强系统文件修改难度。

4) 对于 Linux 进程不受保护，将操作系统内核代码重写，进程监控软件写进内核、或者使用进程监控软件来增加其安全性，现在很多防火墙就拥有这种功能。

5) 对于系统管理员滥用管理权威胁，主要应对措施是加强对系统管理员的培训与监督管理。

3、引入 SSL 协议的重要性：

由以上几点看出一旦主机或者服务器被不法分子控制，他们就可以肆无忌惮随意修改文件、恶意运行某些进程（带有攻击性或者占用非常大的内存）等。不法分子可以通过中间人攻击等一些手段来获取管理员权限控制主机。而 SSL 协议能够有效的解决这些问题，所以对于 SSL 协议的探究显得十分有必要。

2. 2SSL 协议

安全套接层（SSL）是一种计算机网络协议，用于通过不安全的网络（如互联网）来保护网络应用程序客户端和服务端之间的连接。

安全套接字层（SSL）是在 1999 年之前采用 TLS（传输层安全性）之前，最广泛部署的加密协议，用于提供互联网通信的安全性。尽管 SSL 协议已被弃用，并且 TLS 的采用大多数人们仍将这种技术称为“SSL”。

SSL 在通过互联网或内部网络运行的两台机器或设备之间提供安全通道。一个常见的例子是使用 SSL 来保护 Web 浏览器和 Web 服务器之间的通信。这将网站的地址从 HTTP 转为 HTTPS，“S”代表“安全”。

HTTP 是不安全的，并且受到窃听攻击，因为从 web 浏览器传输到 Web 服务器或其他端点之间的数据是以明文形式发送的。这意味着攻击者可以拦截和查看敏感数据，例如信用卡详细信息和帐户登录。当使用 HTTPS 通过浏览器发送或发布数据时，SSL 可以确保这些信息被加密和安全地拦截

SSL 协议为各种高层协议提供基本的安全服务^{[21][22][23]}。

SSL 协议为应用访问连接提供认证、加密和防篡改的服务^{[21][24][25]}。在客户端与服务端文件传输过程中，我们可以通过 SSL 协议将文件数据进行加密，保证了文件传输过程的稳定与安全。正因为它这一特性，消除了用户在网络上数据晚间安全传输方面的顾虑，因此 SSL 一提出，就得到全世界饱受好评。

2.2.1 SSL 协议组成

SSL 协议由记录协议和控制协议构成^{[24][26]}。下层的记录协议可以封装上层发送和接收的数据信息，控制协议的数据。上层的控制协议能够较好地保证网络通信的稳定通畅与安全，可以供身份认证、数据加密等服务。上层协议包括握手协议、错误警报协议、密钥交换协议、记录协议四种。

(1) 握手协议

SSL 握手协议被封装在记录协议中，该协议允许服务端与客户端在传输和接收数据之前互相认证、协商加密算法和密钥。

SSL 协议的握手过程:客户端向服务端发送一些通讯所需要的信息，这些信息包括 SSL 的版本号，加密算法的方式，随机数等一些信息。与此同时，服务端对客户端进行反馈，不仅发送客户端给服务端发送的那些信息，还可以选择发送自己的证书。

这时客户端便会验证这些信息，就会产生两种情况。第一种：就是信息相同，比如 SSL 版本号，加密算法的方式，等一些其他的的信息都相同时，服务端和客户端便可以建立连接。当然还有一种情况是信息不相同，比如 SSL 版本号不同，则需要更新版本。比如加密算法方式不同，服务端无法解读客户端发来的信息，或者比如说证书过期，这些都会导致会话断开连接。

当建立连接之后，客户端便将自己的文件数据通过加密算法（公钥）进行加密，公钥在证书里可以获得，同时产生一个随机数，然后将所有的信息包括公钥全部都发送到服务端。

此时服务端可以选择是否验证客户端。第一种情况是不验证，那么客户端就可以把所有信息全部传送到服务端了。第二种是服务端验证客户端的身份，便会去检查证书和公钥，证书是否合法，是否有效，是否可靠，是否在有效期之内，公钥在证书里，是否可以用来解密文件数据。如果验证不通过便直接断开连接，如果验证通过便可以成功建立连接，这样客户端与服务端便可以成功的通信。

(2) 错误警报协议

客户端和服务端发生报错，便会互相发送劲爆信息。若报错比较严重，便会断开会话连接。，

警报级别:警报级、致命错误级。

(3) 密钥交换协议

密钥交换协议用于通知对方更改加密算法和加密密钥。每次加密协议时，发送密钥交换消息。当发送方发送密钥交换消息，接收方接收密钥交换消息时，必

须及时更改其加密规范，以启动新的加密传输过程。

(4) 记录协议

SSL 协议将所有的传输数据加密封装在记录协议中。在客户机和服务器成功建立会话连接后，客户端和服务端对双方对加密算法的密钥进行交换，解密后进入 SSL 记录协议，读取数据。记录协议向 SSL 连接提供两个服务：

- ①保密性：使用握手协议定义的秘密密钥实现
- ②完整性：握手协议定义了 MAC，用于保证消息完整性

2.2.2 SSL 协议的应用

SSL 协议在 Internet 上有着广泛的应用。虽然它的信道加密协议原理十分简单，没有应用层所具有得抗信用和抗篡改能力，但由于其透明性、简单性、易布局性等特点，在很多领域用的还是非常广泛的，如电子商务。

在电子商务中，网上支付，会有客户、商家和银行三方参与交易。客户先到商家买东西，报到信用卡号，业务查询到银行信用卡信息，信用卡可以继续交易。在这个过程中，可以使用 SSL 协议来保证数据传输的安全性。

客户向企业和银行提供个人信息，是高度相信银行和企业不会泄露或挪用客户资源不。业务与银行之间传输客户数据，验证对方的数字 ID，使安全性得到进一步保障。商家和客户之间的匿名沟通方式，兼顾安全性和易用性。在采用先进的电子协议之前，这种商业模式为在线支付提供了基本的安全保护。

2.2.3 OpenSSL 简介

OpenSSL 是一个密码库，且十分强大，拥有很多的密码算法，常用的密钥以及证书封装管理功能，它能为很多的应用程序提供测试，还有许许多多别的用处。OpenSSL 通常有如下几个特点：

1) 数据保密性：

信息加密就是把数据文件通过加密算法转换成加密的数据文件，无法轻易读取，因而达到文件保护的目的。通过密钥加密数据，再通过密钥解密。没有密钥就无法解读数据文件从而实现保护文件数据的目的。

2) 数据完整：

加密能保证数据文件的一致性。数据文件在传输过程中无法修改。

3) 安全验证：

加密能保证数据的唯一性，它是把用户的密钥作为他安全验证的标识。

2.2.4 SSL 协议在 OpenSSL 中的的实现

SSL 协议由代码，这部分代码位于 OpenSSL 的 SSL 目录下。SSL 的所有实现

中，都有客户端实现、服务端实现、加密实现、记录协议实现、METHOD 方法、客户端服务端都用到的握手方法实现，以及对外提供的函数实现等。

打开 SSL.h 文件，可以看到 SSL 的主要数据结构。有 SSL_CTX、SSL 和 SSL_SESSION 三种。

1) SSL_CTX 为 SSL 握手过程准备，获得所需的证书、私有密钥、版本等选项。

2) SSL 为 SSL 握手阶段传送数据信息。

3) SSL_SESSION 数据结构保存了主密钥、会话 ID，也能够读写各种密钥。

1、SSL 中的密钥相关信息介绍如下：

1) 预主密钥(Premaster SECRET)

预主密钥由客户端生成，通过服务端的公钥加密发送给服务端。它是主密钥的来源。

2) 主密钥(Master SECRET)

由于客户端和服务端的 Master Key 相同，我们可以由客户端和服务端根据预主密钥、客户端和服务端伪随机数获得主密钥。主密钥在 SESSION 数据结构中，可以生成各种密钥信息。

3) MAC 密钥与对称密钥

通过 Master Key、客户端和服务端伪随机数获得 MAC 密钥与对称密钥。

2、相关主要函数：

1) SSL_accept()/SSL_connect;SSL_accept()函数在服务端运行，SSL_connect 函数在客户端运行，用来完成客户端与服务器端的 SSL 握手功能。

2) char *SSL_alert_desc_string_long(int value):该函数用来查找错误原因。

3) SSL_add_client_CA(SSL *ssl,X509 *x):用来添加客户端的 CA。

4)SSL_CIPHER_description:读取对这个加密套件的描述

5) SSL_check_private_key:对 SSL 结构中的私有密钥进行检查

6)SSL_CIPHER_get_bits:该函数功能是获取加密套件里对称算法的加密长度。

7)SSL_do_handshake:负责进行 SSL 握手。

8) SSL_CIPHER_get_name:这个函数的功能是获取加密套件的名字。

9) SSL_CTX_add_client_CA:是给 SSL_CTX 添加客户端 CA。

10)SSL_clear:这个函数的功能是回收占用的内存。

11) SSL_CTX_add_session(SSL_CTX *ctx, SSL_SESSION *s):往 SSL_CTX 添加会话。

12) SSL_CTX_free:释放内存空间

13) SSL_CTX_check_private_key:这个函数的功能是核查私有密钥。

14) SSL_CTX_get_verify_callback:这个函数的功能是获取验证证书的回调函数。

15) long SSL_CTX_get_timeout(const SSL_CTX *s):这个函数功能是获取超时的时间。

- 16) SSL_get_current_cipher:获得当前的加密套件。
- 17) SSL_CTX_get_verify_depth:这个函数的功能是获取证书验证的 depth。
- 18) SSL_CTX_get_verify_mode:这个函数的功能是获得验证方式。
- 19) SSL_get_fd:获得链接句柄。
- 20) SSL_client/server_method:获取客户端和服务端的方法。
- 21) SSL_get_peer_certificate:这个函数的功能是获取对等方的证书。
- 22) SSL_read:接收数据
- 23) SSL_write:发送数据
- 24) SSL_set_fd:设置 SSL 链接句柄。
- 25) SSL_get_current_compression\SSL_get_current_expansion:获取当前的压缩与解压算法的方法。

3、实现步骤:

SSL 的实现分 SSL 记录层的实现与 SSL 握手层的实现，SSL 记录层实现如图 2.1 所示。

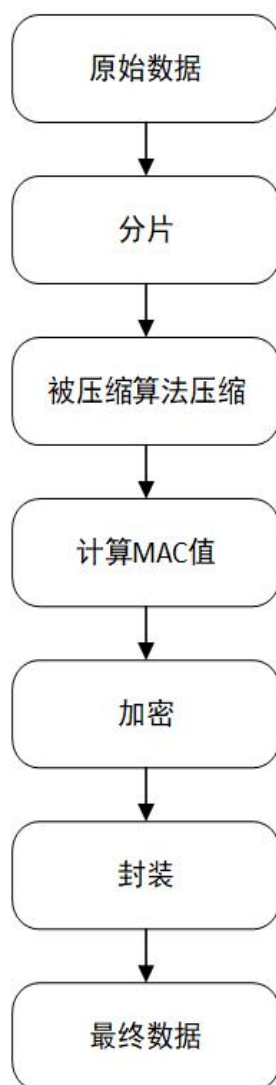


图 2.1 SSL 记录层协议操作过程图

SSL 记录层的实现是:第一步对明文数据进行分片,每片都有固定大小,然后每个分片单独压缩,对每一个压缩包进行 MAC 计算,并产生一个 MAC 值,之后对这个压缩包进行加密,最后将这个加密后的密文与 MAC 值放在一起用协议头进行封装。

SSL 记录层的实现如下:第一步是分离的明文数据成片,每片有一个固定的大小,然后将每个切片单独压缩,并对每个压缩的数据包的 MAC 进行计算,并对这个 MAC 值记录。最后将压缩包进行加密,把加密后的密文和 MAC 值放在一起用协议头进行封装。

SSL 协议握手层实现如图 2.2。SSL 的握手阶段实现如下:

(1) 客户端向服务端发送信息 **hello** 单词以及 SSL 的版本号、加密算法、客户端自动生成的伪随机号（伪随机号一般用软件生成的随机数都是伪随机数，真正的随机数一般需要硬件生成）等，向服务端建立一个新的会话连接。

(2) 服务端根据客户端发过来的信息判断是否可以建立会话连接。如果不可以就无法建立连接，此时客户端会再次发送 **hello** 建立会话连接。由此重复直到建立会话连接或者客户端自动放弃会话连接。如果与服务端会话连接建立，则客户端必须向服务端发送数据加密所需的所有信息，这些信息包括在版本号、支持的加密算法、服务器端自动生成的伪随机数等；

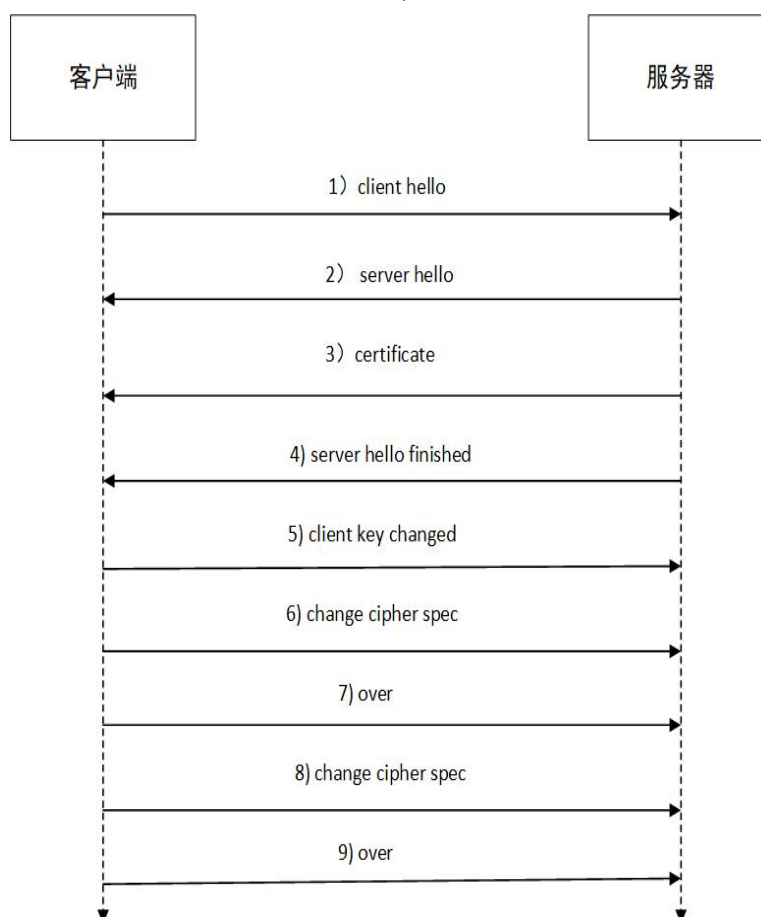


图 2.2 SSL 协议握手层实现过程图

(3)客户端会依据收到的来自服务端的数据产生主 key，并使用服务端的 public key 加密这个主 key 后，将加密过的 key 传给服务端，因为只有服务端有 public key 的另一把 Private key，所以只有真正的服务端能解密；

(4)服务端解密成功后，给客户端返回一段用主 key 认证的数据。

(5)最后双方都要发送 done 或 over 信息表示握手结束。

2.2.5 SSL 协议优缺点分析

SSL 协议操作简单，对网络系统要求的变更不多，而且速度快，成本低，因此在全世界范围内饱受好评。SSL 协议所采用的密钥技术能够保证数据文件的安全传输且不被泄露。在使用过程中，发送方利用接收方的公钥对数据文件进行加密，接收方通过私钥解密读取数据文件。使用 SSL 协议，即使传输过程中，数据文件被非法者盗取，非法者因为没有密钥无法解密读取其中的信息。然而也正是因为 SSL 的加密协议实现的过于简单，容易被不法分子所破解，因此它也有两个主要缺点：

(1)SSL 的保密连接存在较大漏洞：

SSL 协议保证了传输过程的安全性，不能提供任何其他安全保证。客户不知道商家收到信用卡付款是由企业授权的，客户与银行之间缺乏相互认证。非法人可以利用这个漏洞做一些诈骗。比如，在网络上的不法分子假装不熟悉的店铺出现，让消费者相信这些商家，从而进行一些诈骗。即使是一个真实可靠的在线企业，如果你得到了消费者的私人信息，并没有采取保护措施，将很容易被一些互联网黑客抓获。

(2)SSL 协议无法提供很完备的保护好隐私：

在 SSL 交易过程中，消费者需要将所有的支付信息和订单信息都传递给企业。在企业经营过程中可以看到消费者的所有信息，包括信用卡信息。然而消费者不希望企业知悉除了订单信息以外的隐私信息，同样，消费者也希望银行只能看到支付信息，不看其他信息，如所订购东西等。所以在 SSL 交易过程中，客户的隐私无法得到很好的保证。

2.3 文件传输加密算法基础

2.3.1 对称加密算法

对称加密算法的加密与解密是使用同一个密钥算法，使用一个密钥加密解密缺乏安全性。对于某段明文，使用对称加密算法产生一个密钥，并用这个密钥加密，产生的这个密文的长度和原始明文的长度差不多相同。解密时，使用的解密

密钥与加密密钥是同一把密钥。下面详细讨论对称加密算法的各个方面。

1、对称加密算法的几种模式：

1)密码块链模式(Cipher Block Chaining, CBC)

该模式首先将明文分为固定大小的块，然后将最后加密的密文与下一个即将加密的明文块进行加密，然后再用密钥对密文进行加密。第一个明文块是加密的，因为它是第一个数据块，在密文块前面没有加密，所以它需要初始化的向量。它的连接通过明文和密文之间的关系不再相互对应，这样当密文被破解的时候就会比较困难，基本上克服了简单的密文块的传递，即可以被攻击弱点。

2)密码本模式(Electronic Code Book, ECB)

这种模式是最简单的加密模式，也是第一个要在模型中使用的。加密方法是将需要加密的数据分成与密钥长度相同的几个组，最后由同一密钥对每个组进行加密。这种模式的缺点是，它只适用于具有非常小的加密长度的数据，因为这种模式加密所有的数据具有相同的密钥

3)输出反馈模式(Output Feedback Mode, OFB)

这种模式类似于加密反馈模式（CFB），其中的区别在于，后者是充满在加密过程的下一个阶段的密文，而前者，填充的加密过程的下一步是初始化初始化向量加密过程。

4)加密反馈模式(Cipher Feedback Mode, CFB)

使用流加密的加密模式需要使用加密的反馈模式。在这种模式下，应用程序通常为字符流加密。

文献[29-30]分析了在硬件和软件平台上，对算法的特点、复杂度和安全强度进行了比较，得出实现所需资源和实现速度。建议在设计 IPsec 协议时，可以使用对称密钥加密算法，能够有效而又合理地解决安全方案问题。

文献[31-32]分析对称加密算法 AES 和 DES 进行分析对比。提出了一种 AES 加密算法对差分错误进行分析,通过软件模拟来实现根密钥的找回。实验结果表明,只要进行大概 20 次的错误注入,我们就能得到 AES 差分错误分析。同时提出新的一种 DES 差分错误分析方法,这种方法与 AES 的方法在原理上类似,软件模拟的结果说明,破解 DES 需要更多次的错误注入次数。由此我们得出 AES 和 DES 加密算法没有任何的安全保障,很容易受到差分错误分析的攻击,我们可以引入错误侦测机制来抵御此类攻击。

2.3.2 消息摘要加密算法

1、消息摘要算法(Message Digest)概念：

消息摘要算法不管输入多少原始数据，只要进程不发生改变，计算后输出密文长度是基本上不会改变的，如果进程的长度被改变，输出的大小将改变。简单点来说，只要原始数据发生变化，输出将完全不同。该算法计算出的结果不能

逆转，或反转的可能性非常小，通过反求计算难以得到原始类，因此可以作为信息完整性进行验证。当今消息摘要算法主要有安全散列算法（Secure Hash Algorithm, SHA），输出结果为 20 byte，消息摘要算法第五版(Message Digest Algorithm MD_5) 输出结果为 16 byte。消息摘要是把任意长度的输入柔和而产生长度固定的伪随机输入的算法。

2、消息摘要的主要特点有：

(1)消息摘要的长度与输入消息无关。摘要的出处结果位数越长，算法越安全。

(2)消息摘要的产生是伪随机的。不同的输入有不同的输出，且输出唯一的摘要数据。

(3)消息摘要算法的结果是惟一的。也就是说只要输入的信息稍微改变其输出也会改变，可通过这一点来验证数据在传输中是否被篡改。

(4)好的摘要算法，基本上是找不到两条相同的输入信息使得输出摘要相同。

2.4 本章小结

本文首先讲述了越来越多嵌入式应用的系统在 Linux 平台运行。有非常多的互联网企业且会越来越多使用 Linux 平台研发代码。而 Linux 系统存在许多的漏洞，在 Linux 系统上的文件传输便是其中一个，在这里我们就需要引入 SSL 协议。

SSL 协议位于 TCP/IP 协议和各种应用层协议之间，以确保文件的安全传输。SSL 协议可分为两层：SSL 记录协议（SSL 记录协议）在传输协议（如 TCP）上的高级协议，用于数据封装、压缩、加密等功能。SSL 握手协议（SSL 握手协议）建立在 SSL 记录协议上，在实际数据传输开始之前，对双方的通讯双方进行加密算法，交换加密密钥，进行身份认证。

SSL 协议对于数据文件的安全传输发挥着必不可少的作用，但是因为它操作简单，对现有网络系统的修改太少，速度快，成本低等优点在全世界范围内广受好评。然而 SSL 协议也存在着非常多的漏洞，因而对基于 Linux 的文件传输继续探究，引入了加密算法。本文对称加密算法和消息摘要加密算法进行了介绍。

接着本文提出了相对于上两种加密算法改进的 RSA 非对称加密算法，利用线程池多任务处理技术，利用网络传输协议实现在 Linux 平台文件的加密传输过程。

第三章 一种改进的 RSA 加密算法

第二章引入了两种加密算法，分别是对称加密算法和消息摘要加密算法，研究了算法的特点，通过加密方式的复杂化，同时提出了线程池技术，得到了一种改进的 RSA 加密算法，保证了传输过程中密钥的安全性和文件的私密性。

3.1 改进的 RSA 加密算法

通过 RSA 加密算法，Linux 服务器端与客户端的安全加密传输得以实现：

- 1) 在 SSL 协议中获取私钥与公钥（加密文件数据所用到）。
- 2) 在软件设计上建立套接字 Socket，
- 3) 利用 SSL 协议对应用层和 TCP/IP 传输层进行加密，利用过线程池技术实现服务器与客户端多用户的连接、上传与下载等功能。在 Linux 操作系统下，实现客户端与服务器端的文件加密传输。

RSA 算法的核心非对称加密，简单点来说就是加密过程用到公钥和私钥这两种密钥。在客户端与服务器端双方在传输文件之前，接受文件的一方首先将自己密钥对中的公钥送给发送文件的一方，自己持有私钥，用于之后的解密操作。公钥的传输是公开的，即使别人截取了公钥，也没有办法解密经过公钥加密的资料，因为他没有办法获取与该公钥匹配的私钥。

RSA 的安全性是基于两个大素数之积分解的难度。公钥中，值是两个保密的大素数 p 和 q 之积， e 值是 $(p-1)(q-1)$ 的互质数。私钥中的 d 值是根据 e 值和 p, q 的值得到，其中公钥和私钥是以密钥对的形式获取的。位数越大的私钥和公钥，其加解密越有保障。

实现明文数据转化之后的 M 到密文数据 C ，再到明文数据 M 的 RSA 算法的描述如下：

- 1) 设定足够大，不相等的素数 p 和 q ，无人知道 p 和 q 的值；
- 2) 公钥的 n 值： $n=p*q$ ；
- 3) 公钥的 e 值： $f(n)=(p-1)(q-1)$ ，设定一个与 $f(n)$ 互质的数 e ，其中 $1 < e < f(n)$ ；
- 4) 私钥的 d 值： $d = e^{-1} \bmod f(n)$ ；
- 5) 公钥 $KU=(e, n)$ ，私钥 $KR=(d, n)$ ；
- 6) 加密数据：设密文为 C ，则 $C = M^e \pmod n$ ；
- 7) 解密过程为： $M = C^d \pmod n$ 。

本文当中按照这一加密方案设计了 Linux 嵌入式中文件传输的加密过程。在 Linux 操作系统当通过安装 OpenSSL 库来移植 SSL 协议，SSL 协议当中直接实现 RSA 算法。其中 Linux 系统中 OpenSSL 文件包解压后的安装过程如下：

```
$. /config
$make
$make test
$make install
```

所有的 RSA 加密流程都在 SSL 协议当中实现，Linux 系统要实现 RSA 加密，首先就要获取密钥对。提取方法如下：

生成秘钥：

```
OpenSSL genrsa-out privkey. pem 2048
```

此命令会生成 2048 位的密钥 privkey. pem，由于实验的时候用到的数字证书的的申请和颁发机构都是自己，因此用以上私钥来生成与之对应的证书 cacert. pem 生成证书：

```
OpenSSL req-new-x509-key privkey. pem-out cacert. pem-days 1095
```

证书当中包含该私钥对应的公钥。该公钥与以上的私钥是相互匹配的。至此，已经具备了 RSA 加密所需要的密钥对了，之后 RSA 加密解密过程就交给 SSL 来完成，应用该协议之前，主要要熟悉 OpenSSL 的运行机制，以及其中对各种算法的函数过程的描述，这里运用 SSL 协议实现 RSA 数据加密的过程的重要函数部分代码如下：

```
SSL_library_init( ) ; //初始化 SSI 库
OpenSSL_add_all_algorithms( ) ; //载入 SSL 算法
SSL_load_error_strings( ) ; //载入所有错误信息
ctx=SSL_CTX_new(SSLv23_server_method( ) ) ; //创建新的 SSL_
CTX 对象，其中加密方法使用的是 RSA_METHOD
SSL_CTX_use_certificate_file(ctx, temp=strcat(pwd, "/cacert.
pem" ) , SSL_FILETYPE_PEM) //载入数字证书，证书当中包含有公钥
SSL_CTX_use_PrivateKey_file( ctx, temp= strcat (pwd, "/privkey.
pem"), SSL_FILETYPE_PEM) //载入用户私钥
SSL_CTX_check_private_key( ctx) ; //检查私钥是否正确
```

3.2 加密流程

3.2.1 服务端加密流程设计

如图 3.1 所示，服务器端的 SSL 加密流程是：

- 1) 首先对 SSL 库初始化，
- 2) 进行 SSL 算法以及错误信息的载入等初始化工作。
- 3) 在这之后就要进行连接操作了：创建会话连接所使用的协议
- 4) 建立一个 SSL 协议环境 CTX 对象

- 5) 载入包含公钥的数字证书和私钥并且创建套接字 **Socket**，不断检测某一个客户端
- 6) 发出连接申请。如果有客户端正在申请连接，则建立 **SSL** 加密连接
- 7) 根据客户端的菜单选项执行相关的上传下载记录文件等操作
- 8) 完成操作关掉 **SSL** 连接
- 9) 关掉 **Socket**，
- 10) 释放掉 **CTX** 等一些列操作，服务器端又回到初始状态。

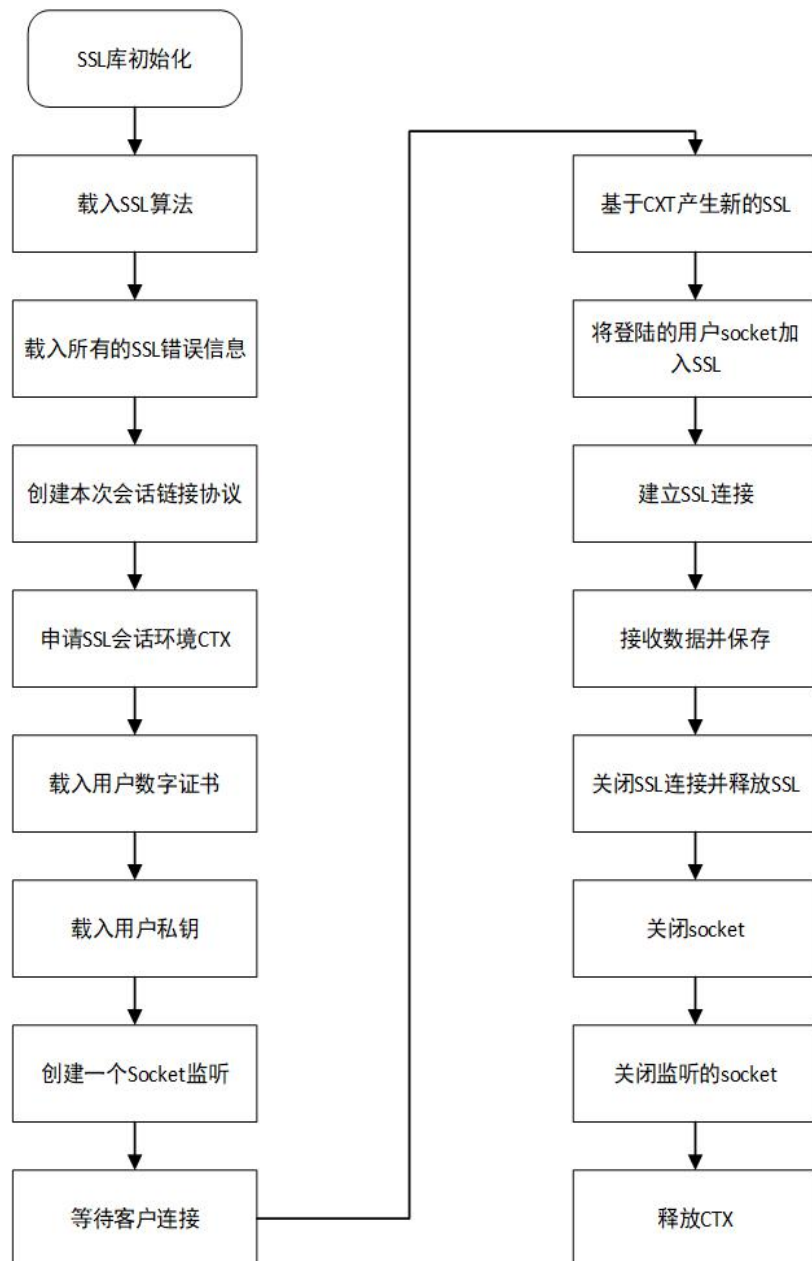


图 3.1 服务端加密流程

3.2.2 客户端加密流程设计

如图 3.2 所示，客户端不需要实现复杂的多个任务的处理程序，也不需要监听多个端口，过程相对简单，客户端与服务器端的加密流程大致是一样的：

- 1) 首先对 SSL 库初始化，
- 2) 进行 SSL 算法以及错误信息的载入等初始化工作。
- 3) 在这之后就要进行连接操作了:创建会话连接所使用的协议
- 4) 建立一个 SSL 协议环境 CTX 对象
- 5) 载入包含公钥的数字证书和私钥并且创建套接字 Socket，不断检测某一个客户端
- 6) 初始化服务器的地址和端口信息
- 7) 发出连接申请。如果有客户端正在申请连接，则建立 SSL 加密连接
- 8) 根据客户端的菜单选项执行相关的上传下载记录文件等操作
- 9) 完成操作关掉 SSL 连接
- 10) 关掉 Socket，
- 11) 释放掉 CTX 等一些列操作，服务器端又回到初始状态。

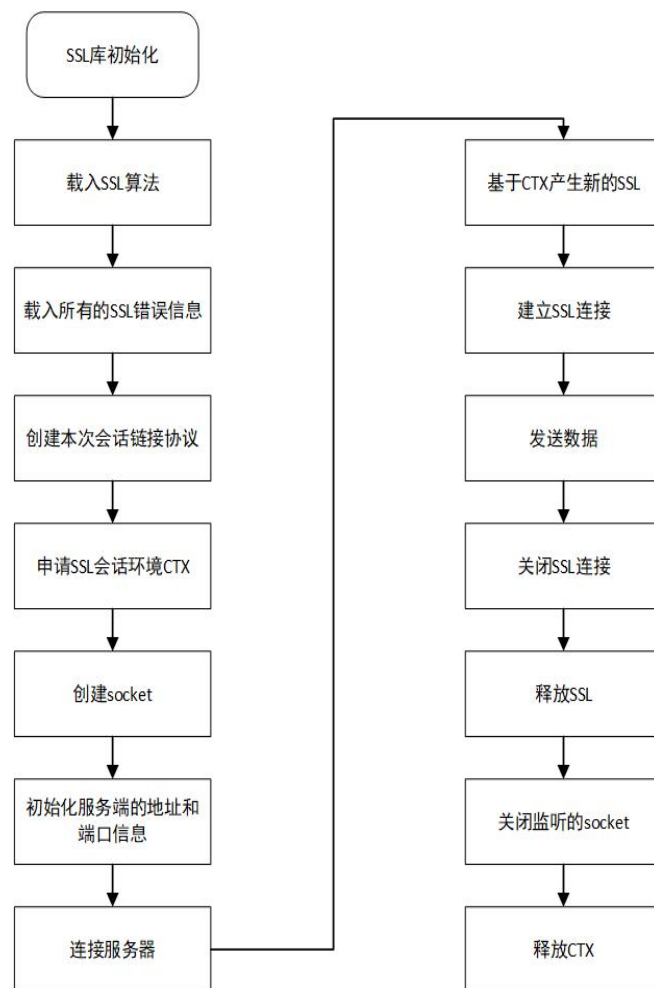


图 3.2 客户端加密

3.2.3 服务器端与客户端交互过程

利用 Linux 的强大的网络编程功能，服务器端和客户端使用的是建立 socket 套接字来实现网络连接，在建立连接的两端建立加密通道即可实现加密传输，服务器端与客户端的交互过程可以用图 3.3 中的流程图来表示：

1) 服务器端的初始化设定服务器的登陆密码，并且读取允许连接的客户端的 IP 地址

2) 当有客户端通过相匹配的 socket 套接字连接到服务器端时，服务器端会将此客户端加入到处处理线程当中

3) 当有另外一个客户端继续连接该服务器，又会有新的线程对该客户端进行处理，只要不超过该服务器允许接人的客户端的最大数量就能处理众多客户端的请求信息。

4) 客户端成功连接到服务器的时候会显示可供设定的菜单栏，根据菜单栏中的命令选项选择相应的下载上传等操作

5) 该图以服务器端与单个客户端的连接阐述了它们之间的交互过程，该过程主要演示的是 socket 套接字的连接过程和宏观的文件加密通道传输交互过程

6) 多个客户端的连接类比此图，当有其他客户端需要连接服务器时，服务器端监听阶段可以监听到的同一网段的匹配的用户名和密码的客户端用户，进行再次连接。

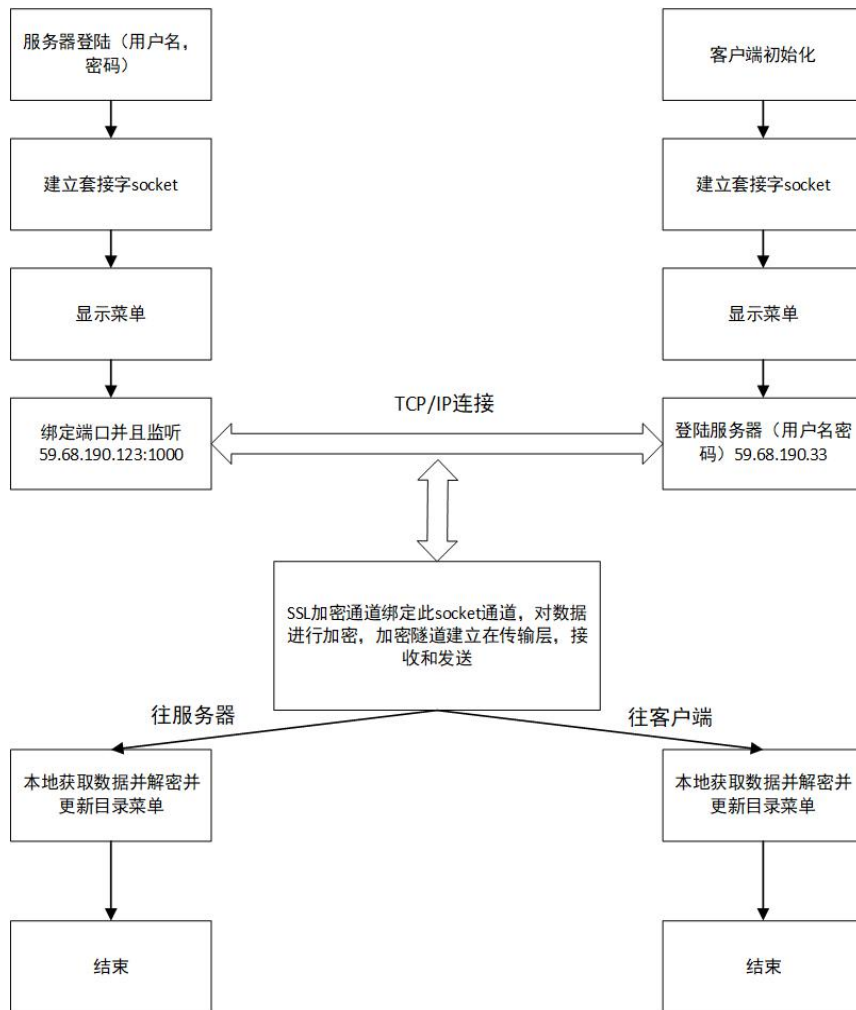


图 3.3 服务端客户端交互过程

3.2.4 传输及交互过程（一对一）

传输的过程需要客户端连接服务器端来进行实时的操作，服务器端要及时快速的响应多个客户端的连接操作，处理多个通道的数据加密传输，由于任务比较多，这一过程会用到线程池来处理多任务操作，用以实现服务器与客户端的一对多互连，双方之间的上传，下载等功能。

3.2.5 引入线程池（多对多）

线程是进程里面处理多任务的一种机制，它用到的地址空间和数据都是和生成这个线程的进程共享的。通过创建线程池而不去创建多进程能节约资源并且能

迅速的对客户端的请求做出反应，在此设计当中线程池的管理用到了条件变量。条件变量能实现线程的阻塞和唤醒工作。具体工作流程是:当线程池在处理任务时，会建立队列，并将任务放入其中。创建线程后这些任务自动启动。如果要处理的任务超过了能处理的最大任务数那么后续的任务会在队列中执行等待操作，等到队列头部的任务完成就会唤醒队列之后的新任务，这样有利于提高工作效率，把 CPU 的闲时工作时间充分利用起来。

本文实现的多任务就是服务器端处理多个客户端任务操作，在服务器端程序中设计线程池，快速的响应多个客户端，实现一对多的文件加密传输通道的设计工作。

用条件变量来实现唤醒机制，也就是说一旦满足某一个条件，就调用互斥锁函数去锁住某个资源，进而执行某一线程，此时只有该线程可以改变该资源的值，其他线程等待该线程执行完。也就是说客户端连接以后，当此客户端正在执行上传某一文件时候，其他的客户端要等待该文件上传完成才能操作，但是其他客户端可以操作此文件以外的其他文件。这样既保证了连接的多通道处理功能，又保证了文件的正确处理，充分利用了 CPU 的空间时间，快速响应多个客户端的操作。

算法分析

3. 3RSA 算法分析

RSA 算法的核心是非对称加密，简单点来说就是加密过程都用到公钥和私钥这两种密钥。在客户端与服务器端双方在传输文件之前，接受文件的一方首先将自己密钥对中的公钥送给发送文件的一方，自己持有私钥，用于之后的解密操作。公钥的传输是公开的，密钥是不需要传输的，即使别人截取了公钥，也没有办法解密经过公钥加密的资料，因为他没有办法获取与该公钥匹配的私钥。但由于这种非对称加密算法复杂，因而加密与解密过程就会变得十分缓慢，会过分的消耗系统资源。

3. 4 本章小结

本文利用 RSA 非对称加密算法，利用线程池多任务处理技术。利用网络传输协议实现在 Linux 平台文件的加密传输过程。其中非对称 RSA 加密算法系基于世界数学难题提出的加密算法且私钥独立持有，使之难以被黑客破解。

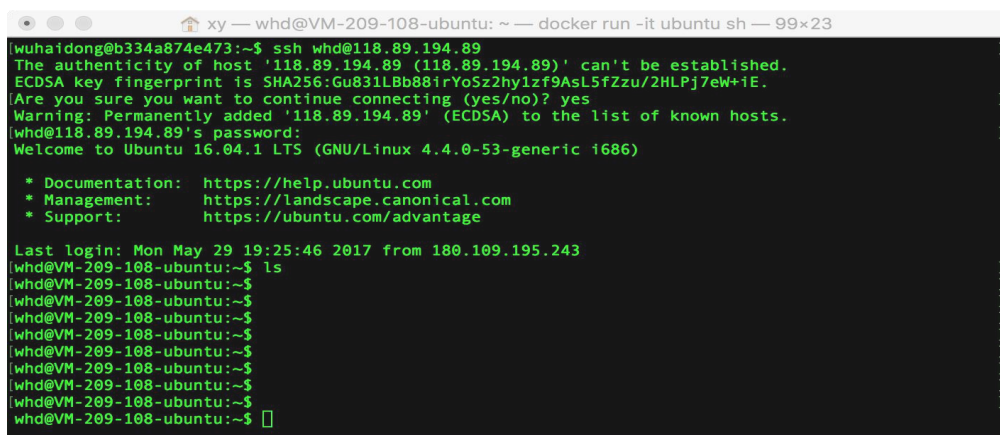
嵌入式技术的发展和 Linux 系统网络传输技术的普及会产生更多的网络安全问题，在现有成熟的网络传输协议的基础上，通过增加加密管道对数据进一步加密处理，能将加密过程在需求更高的嵌入式系统得以运用，十分具备实际应用价值。

第四章 非对称加密算法的实验结果

上文深入研究了 RSA 加密算法，下面是对这种加密算法下，文件的传输的实用性。

4.1 登录验证与连接测试

通过 ssh 协议远程登陆 whd@118.89.194.89，如图 4.1 所示输入密码即可正常登录服务器。



```
wuhaidong@b334a874e473:~$ ssh whd@118.89.194.89
The authenticity of host '118.89.194.89 (118.89.194.89)' can't be established.
ECDSA key fingerprint is SHA256:Gu831LBb881rYoSz2hy1zf9AsL5fZzu/2HLPj7eW+iE.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '118.89.194.89' (ECDSA) to the list of known hosts.
whd@118.89.194.89's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-53-generic i686)

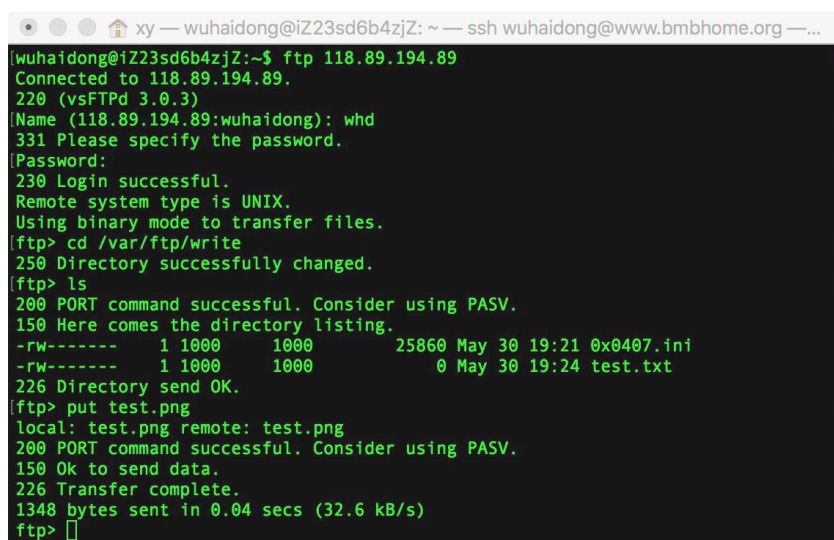
 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Last login: Mon May 29 19:25:46 2017 from 180.109.195.243
whd@VM-209-108-ubuntu:~$ ls
whd@VM-209-108-ubuntu:~$
whd@VM-209-108-ubuntu:~$
whd@VM-209-108-ubuntu:~$
whd@VM-209-108-ubuntu:~$
whd@VM-209-108-ubuntu:~$
whd@VM-209-108-ubuntu:~$
whd@VM-209-108-ubuntu:~$
whd@VM-209-108-ubuntu:~$
whd@VM-209-108-ubuntu:~$
```

图 4.1 登录连接服务器

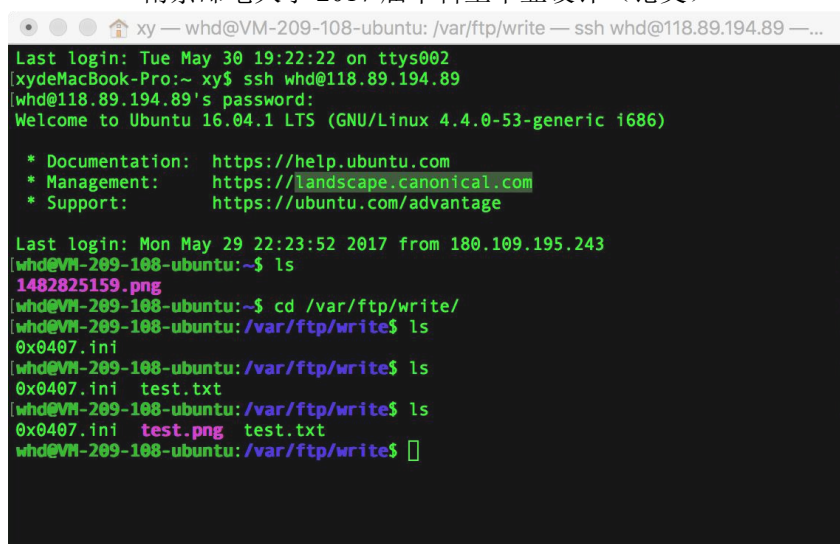
4.2 上传测试

通过 RSA 加密通道，远程上传本地文件到服务器上。如图 4.2 所示，将 test.png 上传到服务器。如图 4.3 所示，文件已经存在于服务器中。



```
wuhaidong@iZ23sd6b4zjZ:~$ ftp 118.89.194.89
Connected to 118.89.194.89.
220 (vsFTPD 3.0.3)
Name (118.89.194.89:wuhaidong): whd
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd /var/ftp/write
250 Directory successfully changed.
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-r--r-- 1 1000 1000 25860 May 30 19:21 0x0407.ini
-rw-r--r-- 1 1000 1000 0 May 30 19:24 test.txt
226 Directory send OK.
ftp> put test.png
local: test.png remote: test.png
200 PORT command successful. Consider using PASV.
150 Ok to send data.
226 Transfer complete.
1348 bytes sent in 0.04 secs (32.6 kB/s)
ftp>
```

图 4.2 文件上传



```

xy — whd@VM-209-108-ubuntu: /var/ftp/write — ssh whd@118.89.194.89 —...
Last login: Tue May 30 19:22:22 on ttys002
[xydeMacBook-Pro:~ xy$ ssh whd@118.89.194.89
whd@118.89.194.89's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-53-generic i686)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

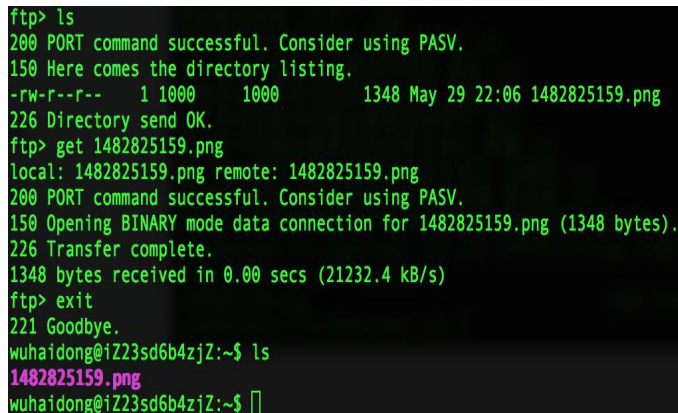
Last login: Mon May 29 22:23:52 2017 from 180.109.195.243
whd@VM-209-108-ubuntu:~$ ls
1482825159.png
whd@VM-209-108-ubuntu:~$ cd /var/ftp/write/
whd@VM-209-108-ubuntu:/var/ftp/write$ ls
0x0407.ini
whd@VM-209-108-ubuntu:/var/ftp/write$ ls
0x0407.ini  test.txt
whd@VM-209-108-ubuntu:/var/ftp/write$ ls
0x0407.ini  test.png  test.txt
whd@VM-209-108-ubuntu:/var/ftp/write$ 

```

图 4.3 文件浏览

4.3 下载

通过 RSA 加密通道，将 1482825159.png 文件下载到本地，先向服务器发送下载指令，在服务器查找文件存在，则允许下载，如图 4.4 文件已经存在于本地。



```

ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-r--r--  1 1000   1000   1348 May 29 22:06 1482825159.png
226 Directory send OK.
ftp> get 1482825159.png
local: 1482825159.png remote: 1482825159.png
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for 1482825159.png (1348 bytes).
226 Transfer complete.
1348 bytes received in 0.00 secs (21232.4 kB/s)
ftp> exit
221 Goodbye.
wuhaidong@iZ23sd6b4zjZ:~$ ls
1482825159.png
wuhaidong@iZ23sd6b4zjZ:~$ 

```

图 4.4 文件下载

4.4 本章小结

实验结果表明数据在 RSA 加密算法的处理下，可以实现文件正常的传输。在现有成熟的网络传输协议的基础上，通过 RSA 加密算法对数据进一步加密处理，具备实际应用价值。

第五章 总结

本文先通过对在 Linux 环境下文件传输进行了研究，发现了安全隐患非常的大，文件的安全对我们的日常生活以及政府军队有着重大的意义，为此我们提出文件安全传输有着重大必要。

SSL 协议是由 SSL 记录协议、握手协议、密钥更改协议和告警协议组成，它们能够认证、加密和篡应用程序的访问连接。SSL 协议指定了在 TCP/IP 协议层间进行数据加密交换，为 TCP/IP 连接提供服务器认证以及客户机认证。SSL 协议在握手过程中通过密钥算法来保证了数据的机密。由于它能够保护敏感信息服务器的安全，消除网民对互联网数据安全的担忧，因此 SSL 在 Internet 上已经被广泛应用。

因为 SSL 协议的文件传输操作比较简单，对现有网络系统没有太多的修改，速度快，成本低等优点而在全世界受到广泛的欢迎。但是它提供的保密连接也有着比较大的漏洞，没有任何的安全保障。在如今互联网大数据成为主流的时代，盗窃者可以轻松通过某种身份暴力破解或者读取个人信息。

因此提出加密算法显得非常有必要。对称加密算法和消息再要算法，非对称加密算法是我的文章中所涉及到的算法，而这其中我们重点研究了 RSA 非对称加密算法。本文通过 RSA 非对称加密算法，利用线程池多任务处理技术。利用网络传输协议实现在 Linux 平台文件的加密传输过程。其中非对称 RSA 加密算法系基于世界数学难题提出的加密算法且私钥独立持有，使之难以被黑客破解。

嵌入式技术的发展和 Linux 系统网络传输技术的普及会产生更多的网络安全问题，在现有成熟的网络传输协议的基础上，通过增加加密管道对数据进一步加密处理，能将加密过程在需求更高的嵌入式系统中得以运用，十分具备实际应用价值。

不过因为是我大学第一次涉及算法，能力有限，再加上时间的局限性，所改进的算法必然存在许多不足之处。首先，算法的研究方法的假设前提是锚节点都能顺利工作，而这种假设在实际操作中不是那么的顺利；其次，仿真模拟实验都是在理想的环境中进行的，与实际应用情况之间有一定的距离。希望在未来的研究生生涯中，我能通过今后的学习与进步对算法进行进一步的修改与补充，以弥补以上不足。

结束语

从 2016 年 12 月至今，我用了大概四个月的时间完成这次的毕业设计。从一开始的对论文写作不知从何下手，到后来懵懵懂懂慢慢找到感觉，再到如今思路清晰，得心应手，整个过程有种苦尽甘来的感觉。历时了这几个月的苦战，毕业设计在紧张而又充实中完成了它的终章。回忆往昔，感慨万千，这段历程带给我带来无数的回忆和收获！

不过由于本人第一次涉及算法，能力有限，再加上时间的局限性，所改进的算法必然存在许多不足之处。首先，算法的研究方法的假设前提是锚节点都能顺利工作，而这种假设在实际操作中不是那么的顺利；其次，仿真模拟实验都是在理想的环境中进行的，与实际应用情况之间有一定的距离。希望在未来的研究生生涯中，我能通过今后的学习与进步对算法进行进一步的修改与补充，以弥补以上不足。

本文利用非对称 RSA 加密算法和线程池多任务处理技术，利用网络传输协议在 Linux 平台实现了服务端和客户端的文件加密传输过程。其中非对称 RSA 加密算法基于世界数学难题突出的加密算法且私钥独立持有，使之难以被黑客破解。同时多线程来处理多客户端任务能保证服务端和客户端的一对多的连接功能以及数据的实时性，且充分保证系统资源能合理分配。

嵌入式技术的发展和 Linux 系统网络传输技术的普及会产生更多的网络安全问题，本文设计的系统能在现有成熟的网络传输协议的基础上，通过增加加密管道对数据进一步加密处理，能将加密过程很方便的移植到安全级别需求高的嵌入式系统当中，具备实际应用价值。

致 谢

本论文在孙老师的认真指导下完成。孙老师是一个知识渊博的人，他对待工作认真负责，眼睛里容不得一粒砂子，指导我们也非常有耐心。孙老师平易近人，在我遇到难题或不熟悉的知识时，孙老师总是孜孜不倦地教导我，让我在写论文的过程中能够克服各种困难。在此，我非常的感谢孙老师。

此外，感谢和我一起度过大学生活的同学们和老师们的陪伴。是你们陪伴我度过了一个又一个夜晚，闯过了一个又一个难关，获得了一次又一次成功.....让我的大学生活丰富多彩。还要感谢管理了我四年的辅导员，不仅关心我们的学习情况，还要关心我们的日常生活。在对待我们的事情上极其认真负责，在此，隆重地对李萌辅导员说声辛苦了。

最后，还要谢谢学校培育了我。学校给我提供了安逸的环境，能让我认认真真的学习，健健康康的成长。

四年的大学生活马上结束，很多人，很多事在我脑海里久久不能忘怀。在这四年中，我褪去了高中的稚嫩，学会了很多做人的道理，亲爱的老师们和可爱的同学们给了我很大的帮助，我的青春回忆中必然会有你们的身影。即将步入社会的我，祝大家一切安好，江湖再见！

参考文献

- [1] Lawrie Brown. Computer Security: Principles and Practices[M]. Prentice Hall, 2007.
- [2] Mark Stamp. Information Security: Principles and Practice[M]. Wiley, 2011.
- [3] Eric Rescorla. SSL and Tls: Designing and Building Secure Systems [M]. Addison-Wesl Professional,2001.
- [4] Douglas E. Comer.用 TC P/IP 进行网际互联—第一卷:原理、协议与结构[M], 第五版. 北京: 电子工业出版社, 2007.
- [5] Douglas E. Comer.用 TC P/IP 进行网际互联—第二卷:设计、实现与内核[M], 第三版. 北京: 电子工业出版社, 2001.
- [6] Stevens W.Richard. TCP/IP 详解一卷一:协议[M]. 北京:机械工业出版社, 2000.
- [7] Linington P.F. File transfer protocols[J]. IEEE Journal on Selected Areas in Communications, 1989,7(7):1052-1059.
- [8] Woodraska Daniel, Sanford Michael, Xu Dianxiang. Security mutation testing of the FileZilla FTP server[A]. 26th Annual ACM Symposium on Applied Computing[C]. Taiwan: Association for Computing Machinery, 2011, 1425 — 1430.
- [9] Brian W.Kernighan, Dennis M.Ritchie. The C programming Language[M]. prentice-Hall, 1988.
- [10] Morioka Y,Higashino T, Tsukamoto K, Komaki S. AP selection algorithm for VoIP services in mixed traffic WLAN environment[A]. Wireless Days, 2008. WD '08. 1st IFIP[C]. Dubai: Conference Publications, 2008, 1-5.
- [11] 刘坚. 基于 MPEG-4 和 RTP 的视频监控系统的研究[D]. 上海交通大学硕士论文, 2005.
- [12] 郭猛. 图像处理管理系统设计与实现[D]. 大连理工大学硕士论文, 2008.
- [13] 何蕴婷, 张宗福. 基于 PKI 的文件安全传输方案研究. 电脑知识与技术 2008, 3(23): 885-887.
- [14] 汪军阳. 有加密功能的文件管理系统的设计与实现[D]. 华中科技大学硕士论文, 2006.
- [15] 刘显强, 林辉, 黄成茂. 基于 TCP 的安全文件传输系统的设计与研究[J]. 计算机光盘软件与应用, 2011(6).
- [16] Wanbo Zheng, Shufen Liu, Zongwei Liu, Qingxing Fu.Security transmission of FTP data based on IPSec[A]. 2009 1st IEEE Symposium on Web Society [C], U.S.: IEEE Computer Society, 2009, 205-208.
- [17] 王科. 基于 IPSec 的 VPN 网关设计及其在电子政务系统中的应用[D]. 武汉理工大学硕士论文, 2010.
- [18] 周国英. IPSec VPN 和 SSL VPN 的协议分析与研究[D]. 太原理工大学硕士论文, 2007.
- [19] Emre Kiciman, Benjamin Livshits. AjaxScope: A Platform for Remotely Monitoring the Client-Side Behavior of Web 2.0 Applications[J]. ACM Transactions on the Web (TWEB), 2010, 4(4).
- [20] Kefei Cheng, Tingqiang Jia, Meng Gao. Research and Implementation of Three HTTPS Attacks[J].Journal of Networks, 2011, 6(5): 757-764.
- [21] Ma Qianqian, Fan Binwen, Sun Yufei and Yang Anli, "Analysis and implementation of audio driver based on ASoC architecture in Linux system," 2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCE C), Xi'an, 2016, pp. 1688-1692.
- [22] J. A. Habibi, R. Munadi and L. V. Yovita, "Analysis secure socket layer protocol with heartbleed bug and distributed denial-of-service," 2016 International Conference on Contr

- ol, Electronics, Renewable Energy and Communications (ICCEREC), Bandung, 2016, pp. 54-59.
- [23] R. K. Jha and F. Khurshid, "Performance analysis of enhanced secure socket layer protocol," 2014 International Conference on Communication and Network Technologies, Sivakasi, 2014, pp. 319-323.
- [24] N. Lim, S. Majumdar and V. Srivastava, "Devising Secure Sockets Layer-based distributed systems: A performance-aware approach," 2012 IEEE 31st International Performance Computing and Communications Conference (IPCCC), Austin, TX, 2012, pp. 376-383.
- [25] K. Kant, R. Iyer and P. Mohapatra, "Architectural impact of secure socket layer on Internet servers," 2012 IEEE 30th International Conference on Computer Design (ICCD), Montreal, QC, 2012, pp. 27-34.
- [26] N. T. Hoa, K. Naoe and Y. Takefuji, "Micro Secure Socket Layer for Micro Server," 2010 Second International Conference on Future Networks, Sanya, Hainan, 2010, pp. 286-290.
- [28] H. Otrok, R. Haraty and A. N. El-Kassar, "Improving the Secure Socket Layer Protocol by modifying its Authentication function," 2006 World Automation Congress, Budapest, 2006, pp. 1-6.
- [29] 杨杰. 3 种对称加密算法在无线闭塞中心中的应用研究[J]. 自动化与仪器仪表, 2017, (04): 135-137.
- [30] 李湘锋, 赵有健, 全成斌. 对称密钥加密算法在 IPsec 协议中的应用[J]. 电子测量与仪器学报, 2014, (01): 75-83.
- [31] 孙维东, 俞军, 沈磊. 对称加密算法 AES 和 DES 的差分错误分析[J]. 复旦学报(自然科学版), 2013, (03): 297-302.
- [32] 姚兰, 姜利群, 李明. 基于双密钥的对称加密算法研究[J]. 微计算机信息, 2008, (21): 56-58.