**2021年暑期生产实习报告**

|  |  |
| --- | --- |
| **学号：** | **U201813742** |
| **班级：** | **信安1804** |
| **姓名：** | **佘新宇** |
| **实习单位：** | **武汉市东西湖区国家网络安全人才与创新基地信息安全实验室** |

**华中科技大学**

**网络空间安全学院**

**目录**

[1 概述 3](#_Toc8064)

[1.1 实习目的 3](#_Toc26644)

[1.2 实习概况 3](#_Toc11887)

[1.3 实习单位及实习岗位简介 3](#_Toc30666)

[1.4 实习计划 4](#_Toc4203)

[2 实习内容 5](#_Toc19102)

[2.1 数据库 5](#_Toc10542)

[2.2 学习gmssl 5](#_Toc22962)

[2.3 国密算法 7](#_Toc18553)

[2.4 实现证书颁发与认证 11](#_Toc3165)

[2.4.1 服务器构成 11](#_Toc7287)

[2.4.2 客户端构成 11](#_Toc29574)

[2.4.3 证书生成 12](#_Toc18486)

[2.4.4 测试服务器构成 12](#_Toc11451)

[2.4.5 客户端构成 13](#_Toc18141)

[2.4.6 证书生成 13](#_Toc4886)

[2.4.7 测试 14](#_Toc18817)

[2.5 实习工作总结 15](#_Toc21548)

[2.4.1 专业知识的应用 15](#_Toc27857)

[2.4.2 实习心得体会 16](#_Toc29088)

[2.4.3 意见、不足及展望 17](#_Toc19694)

[3 实习问题解答 18](#_Toc15367)

# 1 概述

## 实习目的

1. 学习数据库相关知识
2. 学习gmssl相关知识
3. 利用gmssl技术做数据库的证书认证插件

## 实习概况

Mysql/Mariadb数据库mysql.user 的认证默认采用账号+密码方式，其中账号由Host允许登录的客户端ip（%，localhost）和User用户名组成。为增强数据库的认证安全性，可从认证协议入手，设计认证插件，在数据库原有的认证方式基础上使用证书等手段实现数据库客户端与服务端的双向安全认证。

## 实习单位及实习岗位简介

实习地址：武汉市东西湖区国家网络安全人才与创新基地

实习部门：信息安全实验室

实习岗位目标：用C语言实现基于gmssl实现客户端与服务端的证书认证过程

华中科技大学在上世纪80年代就建立了信息安全实验室，在多年积累信息安全教学和研究实践经验的基础上，于2002年建立了信息安全专业，同年设立硕士、博士学位授予点，2003年招收首届本科生和硕士、博士研究生，是国内较早成立本专业的高校之一。2007年成为国内首批“信息安全特色专业”，2016年2月获得首批“网络空间安全”一级学科博士点授予权。

2016年学校积极响应国家战略部署，成立网络空间安全学院。秉执参与国际竞争的历史使命与培养人才的教育使命，学院以建设“一流网络安全学院”为目标，以“德才兼备、软硬兼修、特色强院、学科一流”为办学方针。近几年来，在人才培养方面实施“分级通关”综合实践能力培养方案，是目前国内唯一在本科教学中系统性推行综合实践能力培养的学校。在科研方面，承担了包括973计划项目“云计算安全基础理论与方法研究”、国家自然科学基金重点项目、科技部网络空间安全重点专项课题等多个国家重要项目；在平台建设方面，在依托现有国家、省部级相关平台外，建设有大数据安全湖北省工程研究中心，成立武汉网络安全战略与发展研究院，作为国家网安基地的智库和产业创新源头。在学术交流方面，打造了国内知名品牌的华中科技大学网络空间安全喻园青年科学家论坛，出版网络空间安全技术前沿进展报告；积极承办全国人才培养与学科建设会议以及全国网络空间安全创新能力大赛等，在学界和业界具有较高的声誉。

2017年，华中科技大学与国家网络安全人才与创新基地签订战略合作协议，共同打造极具特色的“网络安全学院+网络安全产业谷”模式，是目前国内唯一的国家网络安全人才与创新基地建设的核心单位之一。位于基地的新院区正按照“21世纪国际一流大学建筑的标杆”目标设计建造，基地顶尖的共享设施与产业园区将为我院的人才培养、科研创新、社会服务等提供优越的发展环境。

华中科技大学是中华人民共和国教育部直属的综合性全国重点大学，中央直管副部级高校，是国家“211工程”重点建设和“985工程”建设高校之一，是首批“双一流”建设高校。建设有武汉光电国家研究中心以及国家脉冲强磁场科学中心（筹）、精密重力测量研究设施等国家重大科技基础设施，还拥有1个国家制造业创新中心、4个国家重点实验室、1个国防科技重点实验室、6个国家工程（技术）研究中心、1个国家临床医学研究中心、1个国家工程实验室、2个国家专业实验室、6个科技部国家国际科技合作基地及一批省部级研究基地。  
华中科技大学2002年建立了信息安全专业，同年设立硕士、博士学位授予点。2007年成为国内首批“信息安全特色专业”，2016年2月获得首批“网络空间安全”一级学科博士点授予权。2018年成立网络空间安全学院，2019年以第一名获批国家一流安全示范学院建设单位（全国只有10个）。入驻国家唯一的网络安全人才创新基地。

## 实习计划

·学习数据库相关知识

·学习openssl、gmssl相关知识

·学习gmssl的C语言函数

·学习gmssl的证书分发与认证

·利用C语言完成证书分发与认证的过程

# **实习内容**

## **数据库**

数据库是存放数据的仓库。它的[存储空间](https://baike.baidu.com/item/%E5%AD%98%E5%82%A8%E7%A9%BA%E9%97%B4/10657950" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93/_blank)很大，可以存放百万条、千万条、上亿条数据。但是数据库并不是随意地将数据进行存放，是有一定的规则的，否则查询的效率会很低。当今世界是一个充满着数据的[互联网](https://baike.baidu.com/item/%E4%BA%92%E8%81%94%E7%BD%91/199186" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93/_blank)世界，充斥着大量的数据。即这个互联网世界就是数据世界。数据的来源有很多，比如出行记录、消费记录、浏览的[网页](https://baike.baidu.com/item/%E7%BD%91%E9%A1%B5/99347" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93/_blank)、发送的消息等等。除了文本类型的数据，图像、音乐、声音都是数据。

数据库是一个按数据结构来存储和管理数据的计算机软件系统。数据库的概念实际包括两层意思：

（1）数据库是一个实体，它是能够合理保管数据的“仓库”，用户在该“仓库”中存放要管理的事务数据，“数据”和“库”两个概念结合成为数据库。

（2）数据库是数据管理的新方法和技术，它能更合适的组织数据、更方便的维护数据、更严密的控制数据和更有效的利用数据。

在数据库的发展历史上，数据库先后经历了层次数据库、网状数据库和关系数据库等各个阶段的发展，数据库技术在各个方面的快速的发展。特别是[关系型数据库](https://baike.baidu.com/item/%E5%85%B3%E7%B3%BB%E5%9E%8B%E6%95%B0%E6%8D%AE%E5%BA%93/8999831" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93/_blank)已经成为目前数据库产品中最重要的一员，80年代以来， 几乎所有的数据库厂商新出的数据库产品都支持关系型数据库，即使一些非关系数据库产品也几乎都有支持关系数据库的接口。这主要是传统的关系型数据库可以比较好的解决管理和存储关系型数据的问题。随着[云计算](https://baike.baidu.com/item/%E4%BA%91%E8%AE%A1%E7%AE%97/9969353" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93/_blank)的发展和[大数据](https://baike.baidu.com/item/%E5%A4%A7%E6%95%B0%E6%8D%AE/1356941" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93/_blank)时代的到来，关系型数据库越来越无法满足需要，这主要是由于越来越多的半关系型和非关系型数据需要用数据库进行存储管理，以此同时，分布式技术等新技术的出现也对数据库的技术提出了新的要求，于是越来越多的非关系型数据库就开始出现，这类数据库与传统的关系型数据库在设计和数据结构有了很大的不同， 它们更强调数据库数据的高并发读写和存储大数据，这类数据库一般被称为[NoSQL](https://baike.baidu.com/item/NoSQL/8828247" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93/_blank)（Not only SQL）数据库。 而传统的关系型数据库在一些传统领域依然保持了强大的生命力。

[数据库系统](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)[DBS](https://baike.baidu.com/item/DBS" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)（Data Base System，简称[DBS](https://baike.baidu.com/item/DBS" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)）通常由[软件](https://baike.baidu.com/item/%E8%BD%AF%E4%BB%B6" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)、[数据库](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)和[数据管理](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E7%AE%A1%E7%90%86" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)员组成。其[软件](https://baike.baidu.com/item/%E8%BD%AF%E4%BB%B6" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)主要包括[操作系统](https://baike.baidu.com/item/%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)、各种[宿主语言](https://baike.baidu.com/item/%E5%AE%BF%E4%B8%BB%E8%AF%AD%E8%A8%80" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)、实用程序以及[数据库管理系统](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)。[数据库](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)由[数据库管理系统](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)统一管理，数据的插入、修改和检索均要通过数据库管理系统进行。[数据管理](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E7%AE%A1%E7%90%86" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)员负责创建、监控和维护整个[数据库](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)，使数据能被任何有权使用的人有效使用。[数据库管理员](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86%E5%91%98" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)一般是由业务水平较高、资历较深的人员担任。

数据库系统的个体含义是指一个具体的数据库管理系统软件和用它建立起来的数据库；它的学科含义是指研究、开发、建立、维护和应用数据库系统所涉及的理论、方法、技术所构成的学科。在这一含义下，数据库系统是软件研究领域的一个重要分支，常称为数据库领域。

[数据库系统](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)是为适应[数据处理](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%A4%84%E7%90%86" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)的需要而发展起来的一种较为理想的数据处理的核心机构。[计算机](https://baike.baidu.com/item/%E8%AE%A1%E7%AE%97%E6%9C%BA" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)的高速处理能力和[大容量存储器](https://baike.baidu.com/item/%E5%A4%A7%E5%AE%B9%E9%87%8F%E5%AD%98%E5%82%A8%E5%99%A8" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)提供了实现[数据管理](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E7%AE%A1%E7%90%86" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)自动化的条件。

[数据库](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)研究跨越于[计算机应用](https://baike.baidu.com/item/%E8%AE%A1%E7%AE%97%E6%9C%BA%E5%BA%94%E7%94%A8" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)、[系统软件](https://baike.baidu.com/item/%E7%B3%BB%E7%BB%9F%E8%BD%AF%E4%BB%B6" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)和理论三个领域，其中应用促进新系统的研制开发，新系统带来新的理论研究，而理论研究又对前两个领域起着指导作用。[数据库系统](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)的出现是[计算机应用](https://baike.baidu.com/item/%E8%AE%A1%E7%AE%97%E6%9C%BA%E5%BA%94%E7%94%A8" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)的一个里程牌，它使得[计算机应用](https://baike.baidu.com/item/%E8%AE%A1%E7%AE%97%E6%9C%BA%E5%BA%94%E7%94%A8" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)从以科学计算为主转向以[数据处理](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%A4%84%E7%90%86" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)为主，并从而使计算机得以在各行各业乃至家庭普遍使用。在它之前的[文件](https://baike.baidu.com/item/%E6%96%87%E4%BB%B6" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)系统虽然也能处理持久数据，但是文件系统不提供对任意部分数据的快速访问，而这对数据量不断增大的应用来说是至关重要的。为了实现对任意部分数据的快速访问，就要研究许多优化技术。这些优化技术往往很复杂，是普通用户难以实现的，所以就由[系统软件](https://baike.baidu.com/item/%E7%B3%BB%E7%BB%9F%E8%BD%AF%E4%BB%B6" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)（[数据库管理系统](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)）来完成，而提供给用户的是简单易用的[数据库语言](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E8%AF%AD%E8%A8%80" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)。由于对[数据库](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)的操作都由[数据库管理系统](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)完成，所以[数据库](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)就可以独立于具体的[应用程序](https://baike.baidu.com/item/%E5%BA%94%E7%94%A8%E7%A8%8B%E5%BA%8F" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)而存在，从而数据库又可以为多个用户所共享。因此，数据的独立性和共享性是[数据库系统](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)的重要特征。数据共享节省了大量人力物力，为[数据库系统](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)的广泛应用奠定了基础。[数据库系统](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)的出现使得普通用户能够方便地将日常数据存入[计算机](https://baike.baidu.com/item/%E8%AE%A1%E7%AE%97%E6%9C%BA" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)并在需要的时候快速访问它们，从而使[计算机](https://baike.baidu.com/item/%E8%AE%A1%E7%AE%97%E6%9C%BA" \t "https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E7%B3%BB%E7%BB%9F/_blank)走出科研机构进入各行各业、进入家庭。

## **学习gmssl**

GmSSL是一个开源的密码工具箱，支持SM2/SM3/SM4/SM9/ZUC等国密(国家商用密码)算法、SM2国密数字证书及基于SM2证书的SSL/TLS安全通信协议，支持国密硬件密码设备，提供符合国密规范的编程接口与命令行工具，可以用于构建PKI/CA、安全通信、数据加密等符合国密标准的安全应用。GmSSL项目是[OpenSSL](https://www.openssl.org/)项目的分支，并与OpenSSL保持接口兼容。因此GmSSL可以替代应用中的OpenSSL组件，并使应用自动具备基于国密的安全能力。GmSSL项目采用对商业应用友好的类BSD开源许可证，开源且可以用于闭源的商业应用。

**关键特性**

·支持SM2/SM3/SM4/SM9/ZUC等全部已公开国密算法

·支持国密SM2双证书SSL套件和国密SM9标识密码套件

·高效实现在主流处理器上可完成4.5万次SM2签名

·支持动态接入具备SKF/SDF接口的硬件密码模块

·支持门限签名、秘密共享和白盒密码等高级安全特性

·支持[Java](http://gmssl.org/docs/java-api.html)、[Go](http://gmssl.org/docs/go-api.html)、[PHP](http://gmssl.org/docs/php-api.html)等多语言接口绑定和REST服务接口

**路线图**

更快、更小、更安全是下一个大版本升级（GmSSL v3.0）的主要目标，我们将从下列方向进行改进：

·采用CMake替代目前基于Perl的构建系统

·支持Linux/Windows/macOS/Android/iOS等主流操作系统，移除对嵌入式OS等其他系统的支持

·支持X86/ARM/RISC-V，针对上述平台64位指令集做汇编层面的优化

·将C语言标准由目前的C89更新为最新的C99或C11，及部分GCC特性，移除对Perl的依赖

·移除不安全的算法和协议，仅支持国密算法和主流国际算法，提升对AEAD、TLS 1.3等新标准的默认支持力度

·提升密码算法抗木马、抗侧信道攻击的安全性

·降低运行时堆内存的使用量，降低总体二进制代码体积

·提供特定于国密算法和协议的统一的多语言（支持Rust/Java/Go/PHP）封装

·保持和OpenSSL最新版本的兼容性，实现GmSSL和OpenSSL在同一个软件中的共存

**国密算法**

国密算法是国家商用密码算法的简称。自2012年以来，国家密码管理局以《中华人民共和国密码行业标准》的方式，陆续公布了SM2/SM3/SM4等密码算法标准及其应用规范。其中“SM”代表“商密”，即用于商用的、不涉及国家秘密的密码技术。其中SM2为基于椭圆曲线密码的公钥密码算法标准，包含数字签名、密钥交换和公钥加密，用于替换RSA/Diffie-Hellman/ECDSA/ECDH等国际算法；SM3为密码哈希算法，用于替代MD5/SHA-1/SHA-256等国际算法；SM4为分组密码，用于替代DES/AES等国际算法；SM9为基于身份的密码算法，可以替代基于数字证书的PKI/CA体系。通过部署国密算法，可以降低由弱密码和错误实现带来的安全风险和部署PKI/CA带来的开销。

**子项目**

本项目包括一些相对独立的子项目：

·[国密浏览器](http://gmssl.org/docs/browser.html)：基于Chromium的支持国密SSL和证书的通用浏览器

·[SDF ENGINE](http://gmssl.org/docs/sdf-engine.html)：支持提供SDF接口的PCI-E密码卡和服务器密码机

·[SKF ENGINE](http://gmssl.org/docs/skf-engine.html)：支持提供SKF接口的USB-KEY、TF卡等智能密码钥匙

·[SM9密钥服务](http://gmssl.org/docs/sm9-pkg.html)：为个人用户提供基于电子邮件地址的SM9标识密钥服务

·[自助CA服务](http://gmssl.org/docs/ca.html)：提供自助式的网站DV证书服务和S/MIME邮件证书服务

·[国密标准文本](https://github.com/guanzhi/GM-Standards)：提供全面的国密GM/T系列标准文本

## **国密算法**

国密算法是国家商用密码算法的简称。自2012年以来，国家密码管理局以《中华人民共和国密码行业标准》的方式，陆续公布了SM2/SM3/SM4等密码算法标准及其应用规范。其中“SM”代表“商密”，即用于商用的、不涉及国家秘密的密码技术。其中SM2为基于椭圆曲线密码的公钥密码算法标准，包含数字签名、密钥交换和公钥加密，用于替换RSA/Diffie-Hellman/ECDSA/ECDH等国际算法；SM3为密码哈希算法，用于替代MD5/SHA-1/SHA-256等国际算法；SM4为分组密码，用于替代DES/AES等国际算法；SM9为基于身份的密码算法，可以替代基于数字证书的PKI/CA体系。通过部署国密算法，可以降低由弱密码和错误实现带来的安全风险和部署PKI/CA带来的开销。

国密算法是我国自主研发创新的一套数据加密处理系列算法。从SM1-SM4分别实现了对称、非对称、摘要等算法功能。特别适合应用于嵌入式物联网等相关领域，完成身份认证和数据加解密等功能。当然，默认的前提条件是算法密钥必须保证安全性，因此要将国密算法嵌入到硬件加密芯片中结合使用。

国密即国家密码局认定的国产密码算法，即商用密码。国密算法是国家密码局制定标准的一系列算法。其中包括了对称加密算法，椭圆曲线非对称加密算法，杂凑算法。具体包括SM1,SM2,SM3等，其中：  
 SM2为国家密码管理局公布的公钥算法，其加密强度为256位。其它几个重要的商用密码算法包括：  
 SM1，对称加密算法，加密强度为128位，采用硬件实现；  
 SM3，密码杂凑算法，杂凑值长度为32字节，和SM2算法同期公布，参见《国家密码管理局公告（第 22 号）》；  
 SMS4，对称加密算法，随WAPI标准一起公布，可使用软件实现，加密强度为128位。

商用密码，是指能够实现商用密码算法的加密、解密和认证等功能的技术。（包括密码算法编程技术和密码算法芯片、加密卡等的实现技术）。商用密码技术是商用密码的核心，国家将商用密码技术列入国家秘密，任何单位和个人都有责任和义务保护商用密码技术的秘密。商用密码的应用领域十分广泛，主要用于对不涉及国家秘密内容但又具有敏感性的内部信息、行政事务信息、经济信息等进行加密保护。比如：商用密码可用于企业门禁管理、企业内部的各类敏感信息的传输加密、存储加密，防止非法第三方获取信息内容；也可用于各种安全认证、网上银行、数字签名等。

例如：在门禁应用中，采用SM1算法进行身份鉴别和数据加密通讯，实现卡片合法性的验证，保证身份识别的真实性。 安全是关系国家、城市信息、行业用户、百姓利益的关键问题。国家密码管理局针对现有重要门禁系统建设和升级改造应用也提出指导意见，加强芯片、卡片、系统的标准化建设。截止目前，国密门禁系统的升级的案例也逐渐增多，基于自主国产知识产权的CPU卡、CPU卡读写设备及密钥管理系统广泛受到关注。一些厂商如同方锐安在2009年推出CPU卡安全门禁系列产品，在2010年北京安博会上，该公司再次向业界展示出“御”系列CPU卡门禁系统、TF-DF6000系列安全门禁读卡器以及基于CPU卡技术的一卡通系统等主流产品和系统。这些厂商是全国推广的国密门禁产品的先驱者，使“御”系列CPU卡门禁系统广泛应用于政府、监狱、司法、军工企业和大型公共智能建筑等高安全领域。

国密算法由国家密码局发布，包含SM1\ SM2\ SM3\ SM4\ SSF33算法;国际算法由美国的安全局发布，是现今最通用的商用算法。

以分组密码算法(DES和SM4)、公钥密码算法(RSA和SM2)、摘要算法(SM3)为例，说一下国密与国际上用的加密算法的区别与联系。

**分组密码算法**：

DES算法是在美国NSA(国家安全局)资助下由IBM公司开发的密码算法，其初衷是为政府非机密的敏感信息提供较强的加密保护。它是美国政府担保的第一种加密算法，并在1977年被正式作为美国联邦信息处理标准。DES主要提供非军事性质的联邦政府机构和私营部门使用，并迅速成为名声最大，使用最广的商用密码算法。

　而在2006年我国公布了无限局域网产品使用的SM4密码算法。这是我国第一次公布自己的商用密码算法。从算法上看，国产SM4算法在计算过程中增加非线性变换，理论上能大大提高其算法的安全性，并且由专业机构进行了密码分析，民间也对21轮SM4进行了差分密码分析，结论均为安全性较高。  
 **公钥密码算法：**

RSA算法由Rivest、Shamir、Adleman于1978年首次发表，是迄今为止最容易理解和实现的公钥算法，已经受住了多年深入的攻击，其理论基础是一种特殊的可逆模幂运算，其安全性基于分解大整数的困难性。密码分析者攻击RSA体制的关键点在于如何分解该大整数，一旦成功分解，等效于获得秘钥。

　　RSA算法既可用于加密，又可用于数字签名，已得到广泛采用，并被许多标准化组织(如ISO、ITU、IETF和SWIFT等)接纳。目前许多国家标准仍采用RSA算法或它的变型。

SM2算法由国家密码管理局于2010年12月17日发布，全称为椭圆曲线算法。椭圆曲线并不是椭圆，之所以称为椭圆曲线是因为它们是用三次方程来表示的，并且该方程与计算椭圆周长的方程相似。SM2算法的安全性基于一个数学难题”离散对数问题ECDLP”实现。

现今对椭圆曲线研究的时间短，经过许多优秀的数学家的努力，至今一直没有找到亚指数级算法。正是由于目前所知求解ECDLP的最好方法是指数级的，这使得我们选用SM2算法作加解密及数字签名时，所要求的密钥长度比RSA要短得多。

**摘要算法：**

摘要函数在密码学中具有重要的地位,被广泛应用在数字签名,消息认证,数据完整性检测等领域。摘要函数通常被认为需要满足三个基本特性：碰撞稳固性,原根稳固性和第二原根稳固性。

　　2005年,Wang等人给出了MD5算法和SHA-1算法的碰撞攻击方法，现今被广泛应用的MD5算法和SHA-1算法不再是安全的算法。

SM3密码摘要算法是中国国家密码管理局2010年公布的中国商用密码杂凑算法标准。SM3算法适用于商用密码应用中的数字签名和验证，是在SHA-256基础上改进实现的一种算法。SM3算法采用Merkle-Damgard结构，消息分组长度为512位，摘要值长度为256位。SM3算法的压缩函数与SHA-256的压缩函数具有相似的结构,但是SM3算法的设计更加复杂,比如压缩函数的每一轮都使用2个消息字。现今为止，SM3算法的安全性相对较高。

总的来说，为了保障商用密码的安全性，国家商用密码管理办公室制定了一系列密码标准，包括SM1（SCB2）、SM2、SM3、SM4、SM7、SM9、祖冲之密码算法（ZUC)那等等。其中SM1、SM4、SM7、祖冲之密码（ZUC）是对称算法；SM2、SM9是非对称算法；SM3是哈希算法。目前，这些算法已广泛应用于各个领域中，期待有一天会有采用国密算法的区块链应用出现。

其中SM1、SM7算法不公开，调用该算法时，需要通过加密芯片的接口进行调用。

1. SM1对称密码

SM1 算法是分组密码算法，分组长度为128位，密钥长度都为 128 比特，算法安全保密强度及相关软硬件实现性能与 AES 相当，算法不公开，仅以IP核的形式存在于芯片中。

采用该算法已经研制了系列芯片、智能IC卡、智能密码钥匙、加密卡、加密机等安全产品，广泛应用于电子政务、电子商务及国民经济的各个应用领域（包括国家政务通、警务通等重要领域）。

2. SM2椭圆曲线公钥密码算法

SM2算法就是ECC椭圆曲线密码机制，但在签名、密钥交换方面不同于ECDSA、ECDH等国际标准，而是采取了更为安全的机制。另外，SM2推荐了一条256位的曲线作为标准曲线。

SM2标准包括总则，数字签名算法，密钥交换协议，公钥加密算法四个部分，并在每个部分的附录详细说明了实现的相关细节及示例。

SM2算法主要考虑素域Fp和F2m上的椭圆曲线，分别介绍了这两类域的表示，运算，以及域上的椭圆曲线的点的表示，运算和多倍点计算算法。然后介绍了编程语言中的数据转换，包括整数和字节串，字节串和比特串，域元素和比特串，域元素和整数，点和字节串之间的数据转换规则。

详细说明了有限域上椭圆曲线的参数生成以及验证，椭圆曲线的参数包括有限域的选取，椭圆曲线方程参数，椭圆曲线群基点的选取等，并给出了选取的标准以便于验证。最后给椭圆曲线上密钥对的生成以及公钥的验证，用户的密钥对为（s，sP），其中s为用户的私钥，sP为用户的公钥，由于离散对数问题从sP难以得到s，并针对素域和二元扩域给出了密钥对生成细节和验证方式。总则中的知识也适用于SM9算法。

在总则的基础上给出了数字签名算法（包括数字签名生成算法和验证算法），密钥交换协议以及公钥加密算法（包括加密算法和解密算法），并在每个部分给出了算法描述，算法流程和相关示例。

数字签名算法，密钥交换协议以及公钥加密算法都使用了国家密管理局批准的SM3密码杂凑算法和随机数发生器。数字签名算法，密钥交换协议以及公钥加密算法根据总则来选取有限域和椭圆曲线，并生成密钥对。

SM2算法在很多方面都优于RSA算法（RSA发展得早应用普遍，SM2领先也很自然）。

3. SM3杂凑算法

SM3密码杂凑（哈希、散列）算法给出了杂凑函数算法的计算方法和计算步骤，并给出了运算示例。此算法适用于商用密码应用中的数字签名和验证，消息认证码的生成与验证以及随机数的生成，可满足多种密码应用的安全需求。在SM2，SM9标准中使用。

此算法对输入长度小于2的64次方的比特消息，经过填充和迭代压缩，生成长度为256比特的杂凑值，其中使用了异或，模，模加，移位，与，或，非运算，由填充，迭代过程，消息扩展和压缩函数所构成。具体算法及运算示例见SM3标准。

4. SM4对称算法

此算法是一个分组算法，用于无线局域网产品。该算法的分组长度为128比特，密钥长度为128比特。加密算法与密钥扩展算法都采用32轮非线性迭代结构。解密算法与加密算法的结构相同，只是轮密钥的使用顺序相反，解密轮密钥是加密轮密钥的逆序。

此算法采用非线性迭代结构，每次迭代由一个轮函数给出，其中轮函数由一个非线性变换和线性变换复合而成，非线性变换由S盒所给出。其中rki为轮密钥，合成置换T组成轮函数。轮密钥的产生与上图流程类似，由加密密钥作为输入生成，轮函数中的线性变换不同，还有些参数的区别。SM4算法的具体描述和示例见SM4标准。

5. SM7对称密码

SM7算法，是一种分组密码算法，分组长度为128比特，密钥长度为128比特。SM7适用于非接触式IC卡，应用包括身份识别类应用(门禁卡、工作证、参赛证)，票务类应用(大型赛事门票、展会门票)，支付与通卡类应用（积分消费卡、校园一卡通、企业一卡通等）。

6. SM9标识密码算法

为了降低公开密钥系统中密钥和证书管理的复杂性，以色列科学家、RSA算法发明人之一Adi Shamir在1984年提出了标识密码（Identity-Based Cryptography）的理念。标识密码将用户的标识（如邮件地址、手机号码、QQ号码等）作为公钥，省略了交换数字证书和公钥过程，使得安全系统变得易于部署和管理，非常适合端对端离线安全通讯、云端数据加密、基于属性加密、基于策略加密的各种场合。2008年标识密码算法正式获得国家密码管理局颁发的商密算法型号：SM9(商密九号算法)，为我国标识密码技术的应用奠定了坚实的基础。

SM9算法不需要申请数字证书，适用于互联网应用的各种新兴应用的安全保障。如基于云技术的密码服务、电子邮件安全、智能终端保护、物联网安全、云存储安全等等。这些安全应用可采用手机号码或邮件地址作为公钥，实现数据加密、身份认证、通话加密、通道加密等安全应用，并具有使用方便，易于部署的特点，从而开启了普及密码算法的大门。

7. ZUC祖冲之算法

祖冲之序列密码算法是中国自主研究的流密码算法,是运用于移动通信4G网络中的国际标准密码算法,该算法包括祖冲之算法(ZUC)、加密算法(128-EEA3)和完整性算法(128-EIA3)三个部分。目前已有对ZUC算法的优化实现，有专门针对128-EEA3和128-EIA3的硬件实现与优化。

密码算法作为国家战略资源，比历史上任何时候都显得更为关键。在大数据和云计算的时代，关键信息往往通过数据挖掘技术在海量数据中获得，所以每一个人的信息保护都非常重要。

## **实现证书颁发与认证**

### 2.4.1 服务器构成

·初始化

SSLeay\_add\_ssl\_algorithms()

OpenSSL\_add\_all\_algorithms()

SSL\_load\_error\_strings()

ERR\_load\_BIO\_strings()

·等待接收到连接

Accept()

·使用GMTLS\_client\_method方法

SSL\_CTX\_new()

·加载CA的证书

SSL\_CTX\_load\_verify\_locations()

·加载自己的证书和私钥（包括证书和加密证书）

SSL\_CTX\_use\_certificate\_file()

SSL\_CTX\_use\_PrivateKey\_file()

私钥的作用是，ssl握手过程中，对客户端发送过来的随机消息进行加密，然后客户端再使用服务器的公钥进行解密，若解密后的原始消息跟客户端发送的消息一致，则认为此服务器是客户端想要链接的服务器

·验证私钥正确性

SSL\_CTX\_check\_private\_key()

·将连接交付给SSL并进行SSL连接

SSL\_new()

SSL\_connect()

·进行操作

·释放资源

SSL\_free()

SSL\_CTX\_free()

其中：

SSL\_library\_init()注册可用的SSL / TLS密码和摘要。

OpenSSL\_add\_ssl\_algorithms()和SSLeay\_add\_ssl\_algorithms()是同义词SSL\_library\_init() 。

必须先调用SSL\_library\_init()，然后才能执行其他任何操作。SSL\_library\_init()不可重入。

SSL\_CTX\_new()创建一个新的SSL\_CTX对象作为框架，以建立启用TLS/SSL的连接。

SSL\_CTX对象使用method作为连接方法。这些方法以通用类型(供客户端和服务器使用)，仅服务器类型和仅客户端类型存在。方法可以是以下类型：

SSLv23\_method()，SSLv23\_server\_method()，SSLv23\_client\_method()  
 这些是通用的、版本灵活的SSL/TLS方法。实际使用的协议版本将协商到客户端和服务器相互支持的最高版本。支持的协议有SSLv2、SSLv3、TLSv1、TLSv1.1和TLSv1.2。大多数应用程序都应该使用这些方法，并避免使用下面描述的特定于版本的方法。

函数的SSL\_CTX\_\*类将证书和密钥加载到SSL\_CTX对象ctx中。这些信息通过复制被传递到SSL对象，SSL是通过复制从ctx使用SSL\_new(3)创建的，因此应用于ctx的更改不会传播到已经存在的SSL对象。

函数的SSL\_\*类只将证书和密钥加载到特定的SSL对象中。当为这个SSL对象调用SSL\_clear(3)时，将保留特定的信息。

SSL\_CTX\_use\_certificate()将证书x加载到ctx中，SSL\_use\_certificate()将x加载到ssl中。可以使用SSL\_CTX\_add\_extra\_chain\_cert(3)函数指定形成完整证书链所需的其余证书。

SSL\_CTX\_use\_certificate\_ASN1()将ASN1编码的证书从内存位置d(带长度len)加载到ctx, SSL\_use\_certificate\_ASN1()将ASN1编码的证书加载到ssl。

SSL\_CTX\_use\_certificate\_file()将第一个存储在文件中的证书加载到ctx。证书的格式化类型必须从已知的类型SSL\_FILETYPE\_PEM、SSL\_FILETYPE\_ASN1中指定。SSL\_use\_certificate\_file()将证书从文件加载到ssl。SSL\_CTX\_use\_certificate\_chain\_file()将证书链从文件加载到ctx。证书必须是PEM格式，并且必须从主题的证书(实际的客户端或服务器证书)开始排序，然后是中间的CA证书(如果适用的话)，并在最高级别(根)CA结束。在单个SSL对象上没有相应的函数。

SSL\_CTX\_use\_PrivateKey()将pkey作为私钥添加到ctx。 SSL\_CTX\_use\_RSAPrivateKey()将rsa类型的私钥rsa添加到ctx。 SSL\_use\_PrivateKey()将pkey作为私钥添加到ssl;SSL\_use\_RSAPrivateKey()将rsa作为rsa类型的私钥添加到ssl。如果已经设置了证书，并且私有证书不属于该证书，则返回一个错误。要更改证书，需要使用SSL\_CTX\_use\_certificate()或SSL\_CTX\_use\_certificate()设置新证书的私钥对，然后再使用SSL\_CTX\_use\_PrivateKey()或SSL\_use\_PrivateKey()设置私钥。

SSL\_CTX\_use\_PrivateKey\_ASN1()将存储在内存位置d的pk类型私钥(长度len)添加到ctx。SSL\_CTX\_use\_RSAPrivateKey\_ASN1()将存储在内存位置d的RSA类型的私钥(长度len)添加到ctx。SSL\_use\_PrivateKey\_ASN1()和SSL\_use\_RSAPrivateKey\_ASN1()将私钥添加到ssl。

SSL\_CTX\_use\_PrivateKey\_file()将文件中找到的第一个私钥添加到ctx。证书的格式化类型必须从已知的类型SSL\_FILETYPE\_PEM、SSL\_FILETYPE\_ASN1中指定。

SSL\_CTX\_use\_RSAPrivateKey\_file()将文件中找到的第一个私有RSA密钥添加到ctx。SSL\_use\_PrivateKey\_file()将文件中找到的第一个私钥添加到ssl;SSL\_use\_RSAPrivateKey\_file()将找到的第一个私有RSA密钥添加到ssl。

SSL\_CTX\_check\_private\_key()检查私钥与装载到ctx中的相应证书的一致性。如果安装了多个密钥/证书对(RSA/DSA)，将检查最后安装的项。如果最后一项是RSA证书或密钥，将检查RSA密钥/证书对。SSL\_check\_private\_key()对ssl执行相同的检查。如果没有为此ssl显式添加密钥/证书，则将检查添加到ctx中的最后一项。

SSL\_set\_fd()将文件描述符fd设置为SSL的TLS/SSL(加密)端输入/输出设施。fd通常是网络连接的套接字文件描述符。  
 在执行操作时，将自动创建套接字BIO来连接ssl和fd。BIO和SSL引擎继承fd的行为。如果fd是非阻塞的，ssl也将具有非阻塞行为。

如果已经有一个BIO连接到ssl，那么将调用BIO\_free()(对于读写端，如果不同的话)。  
 SSL\_set\_rfd()和SSL\_set\_wfd()执行各自的操作，但仅针对可独立设置的读通道或写通道。

SSL\_accept()等待TLS/SSL客户机启动TLS/SSL握手。通过设置底层BIO，通信通道必须已经被设置并分配给ssl。

SSL\_connect()启动与服务器的TLS/SSL握手。通过设置底层BIO，通信通道必须已经被设置并分配给ssl。

SSL\_CTX\_free()减少ctx的引用计数，并删除ctx指向的SSL\_CTX对象，如果引用计数达到0，则释放分配的内存。  
 它还调用间接受影响项的free()过程(如果适用的话):会话缓存、密码列表、客户端ca列表、证书和密钥。

### 2.4.2 客户端构成

·初始化

·使用GMTLS\_client\_method方法

·要求校验服务器证书

SSL\_CTX\_set\_verify()

·加载CA证书

·加载自己的证书和私钥（包括证书和加密证书）

·创建连接、连接服务器

Socket()

Connect()

·将连接交付给SSL并进行SSL连接

·进行操作

·释放资源

其中：

SSL\_library\_init()注册可用的SSL / TLS密码和摘要。

OpenSSL\_add\_ssl\_algorithms()和SSLeay\_add\_ssl\_algorithms()是同义词SSL\_library\_init() 。

必须先调用SSL\_library\_init()，然后才能执行其他任何操作。SSL\_library\_init()不可重入。

SSL\_CTX\_new()创建一个新的SSL\_CTX对象作为框架，以建立启用TLS/SSL的连接。

SSL\_CTX对象使用method作为连接方法。这些方法以通用类型(供客户端和服务器使用)，仅服务器类型和仅客户端类型存在。方法可以是以下类型：

SSLv23\_method()，SSLv23\_server\_method()，SSLv23\_client\_method()  
 这些是通用的、版本灵活的SSL/TLS方法。实际使用的协议版本将协商到客户端和服务器相互支持的最高版本。支持的协议有SSLv2、SSLv3、TLSv1、TLSv1.1和TLSv1.2。大多数应用程序都应该使用这些方法，并避免使用下面描述的特定于版本的方法。

函数的SSL\_CTX\_\*类将证书和密钥加载到SSL\_CTX对象ctx中。这些信息通过复制被传递到SSL对象，SSL是通过复制从ctx使用SSL\_new(3)创建的，因此应用于ctx的更改不会传播到已经存在的SSL对象。

函数的SSL\_\*类只将证书和密钥加载到特定的SSL对象中。当为这个SSL对象调用SSL\_clear(3)时，将保留特定的信息。

SSL\_CTX\_use\_certificate()将证书x加载到ctx中，SSL\_use\_certificate()将x加载到ssl中。可以使用SSL\_CTX\_add\_extra\_chain\_cert(3)函数指定形成完整证书链所需的其余证书。

SSL\_CTX\_use\_certificate\_ASN1()将ASN1编码的证书从内存位置d(带长度len)加载到ctx, SSL\_use\_certificate\_ASN1()将ASN1编码的证书加载到ssl。

SSL\_CTX\_use\_certificate\_file()将第一个存储在文件中的证书加载到ctx。证书的格式化类型必须从已知的类型SSL\_FILETYPE\_PEM、SSL\_FILETYPE\_ASN1中指定。SSL\_use\_certificate\_file()将证书从文件加载到ssl。SSL\_CTX\_use\_certificate\_chain\_file()将证书链从文件加载到ctx。证书必须是PEM格式，并且必须从主题的证书(实际的客户端或服务器证书)开始排序，然后是中间的CA证书(如果适用的话)，并在最高级别(根)CA结束。在单个SSL对象上没有相应的函数。

SSL\_CTX\_use\_PrivateKey()将pkey作为私钥添加到ctx。 SSL\_CTX\_use\_RSAPrivateKey()将rsa类型的私钥rsa添加到ctx。 SSL\_use\_PrivateKey()将pkey作为私钥添加到ssl;SSL\_use\_RSAPrivateKey()将rsa作为rsa类型的私钥添加到ssl。如果已经设置了证书，并且私有证书不属于该证书，则返回一个错误。要更改证书，需要使用SSL\_CTX\_use\_certificate()或SSL\_CTX\_use\_certificate()设置新证书的私钥对，然后再使用SSL\_CTX\_use\_PrivateKey()或SSL\_use\_PrivateKey()设置私钥。

SSL\_CTX\_use\_PrivateKey\_ASN1()将存储在内存位置d的pk类型私钥(长度len)添加到ctx。SSL\_CTX\_use\_RSAPrivateKey\_ASN1()将存储在内存位置d的RSA类型的私钥(长度len)添加到ctx。SSL\_use\_PrivateKey\_ASN1()和SSL\_use\_RSAPrivateKey\_ASN1()将私钥添加到ssl。

SSL\_CTX\_use\_PrivateKey\_file()将文件中找到的第一个私钥添加到ctx。证书的格式化类型必须从已知的类型SSL\_FILETYPE\_PEM、SSL\_FILETYPE\_ASN1中指定。

SSL\_CTX\_use\_RSAPrivateKey\_file()将文件中找到的第一个私有RSA密钥添加到ctx。SSL\_use\_PrivateKey\_file()将文件中找到的第一个私钥添加到ssl;SSL\_use\_RSAPrivateKey\_file()将找到的第一个私有RSA密钥添加到ssl。

SSL\_CTX\_check\_private\_key()检查私钥与装载到ctx中的相应证书的一致性。如果安装了多个密钥/证书对(RSA/DSA)，将检查最后安装的项。如果最后一项是RSA证书或密钥，将检查RSA密钥/证书对。SSL\_check\_private\_key()对ssl执行相同的检查。如果没有为此ssl显式添加密钥/证书，则将检查添加到ctx中的最后一项。

SSL\_set\_fd()将文件描述符fd设置为SSL的TLS/SSL(加密)端输入/输出设施。fd通常是网络连接的套接字文件描述符。  
 在执行操作时，将自动创建套接字BIO来连接ssl和fd。BIO和SSL引擎继承fd的行为。如果fd是非阻塞的，ssl也将具有非阻塞行为。

如果已经有一个BIO连接到ssl，那么将调用BIO\_free()(对于读写端，如果不同的话)。  
 SSL\_set\_rfd()和SSL\_set\_wfd()执行各自的操作，但仅针对可独立设置的读通道或写通道。

SSL\_accept()等待TLS/SSL客户机启动TLS/SSL握手。通过设置底层BIO，通信通道必须已经被设置并分配给ssl。

SSL\_connect()启动与服务器的TLS/SSL握手。通过设置底层BIO，通信通道必须已经被设置并分配给ssl。

SSL\_CTX\_free()减少ctx的引用计数，并删除ctx指向的SSL\_CTX对象，如果引用计数达到0，则释放分配的内存。  
 它还调用间接受影响项的free()过程(如果适用的话):会话缓存、密码列表、客户端ca列表、证书和密钥。

### 2.4.3 证书生成

该部分在inership/mk\_tls\_cert/generate.sh中写的很详细，每一步都有说明这里仅给出gmssl req的部分参数信息

# gmssl req 创建证书签名请求等功能

# -nodes 对私钥不进行加密,不用手动输入密码

# -newkey 创建CSR证书签名文件和RSA私钥文件

# rsa:2048 指定创建的RSA私钥长度为2048

# -keyout 创建的私钥文件名称

# -out 指定CSR输出文件名

# -subj 指定证书Subject内容

# Subject设定内容说明

#

# 字段 含义 设定值例

# /C= Country CN

# /ST= State HUBEI

# /L= Location WUHAN

# /O= Organization HUST

# /OU= Organizational CA

# /CN= Common Name sxy

### 2.4.4 测试服务器构成

·初始化

SSLeay\_add\_ssl\_algorithms()

OpenSSL\_add\_all\_algorithms()

SSL\_load\_error\_strings()

ERR\_load\_BIO\_strings()

·等待接收到连接

Accept()

·使用GMTLS\_client\_method方法

SSL\_CTX\_new()

·加载CA的证书

SSL\_CTX\_load\_verify\_locations()

·加载自己的证书和私钥（包括证书和加密证书）

SSL\_CTX\_use\_certificate\_file()

SSL\_CTX\_use\_PrivateKey\_file()

私钥的作用是，ssl握手过程中，对客户端发送过来的随机消息进行加密，然后客户端再使用服务器的公钥进行解密，若解密后的原始消息跟客户端发送的消息一致，则认为此服务器是客户端想要链接的服务器

·验证私钥正确性

SSL\_CTX\_check\_private\_key()

·将连接交付给SSL并进行SSL连接

SSL\_new()

SSL\_connect()

·进行操作

·释放资源

SSL\_free()

SSL\_CTX\_free()

### 2.4.5 客户端构成

·初始化

·使用GMTLS\_client\_method方法

·要求校验服务器证书

SSL\_CTX\_set\_verify()

·加载CA证书

·加载自己的证书和私钥（包括证书和加密证书）

·创建连接、连接服务器

Socket()

Connect()

·将连接交付给SSL并进行SSL连接

·进行操作

·释放资源

### 2.4.6 证书生成

该部分在inership/mk\_tls\_cert/generate.sh中写的很详细，每一步都有说明这里仅给出gmssl req的部分参数信息

# gmssl req 创建证书签名请求等功能

# -nodes 对私钥不进行加密,不用手动输入密码

# -newkey 创建CSR证书签名文件和RSA私钥文件

# rsa:2048 指定创建的RSA私钥长度为2048

# -keyout 创建的私钥文件名称

# -out 指定CSR输出文件名

# -subj 指定证书Subject内容

# Subject设定内容说明

#

# 字段 含义 设定值例

# /C= Country CN

# /ST= State HUBEI

# /L= Location WUHAN

# /O= Organization HUST

# /OU= Organizational CA

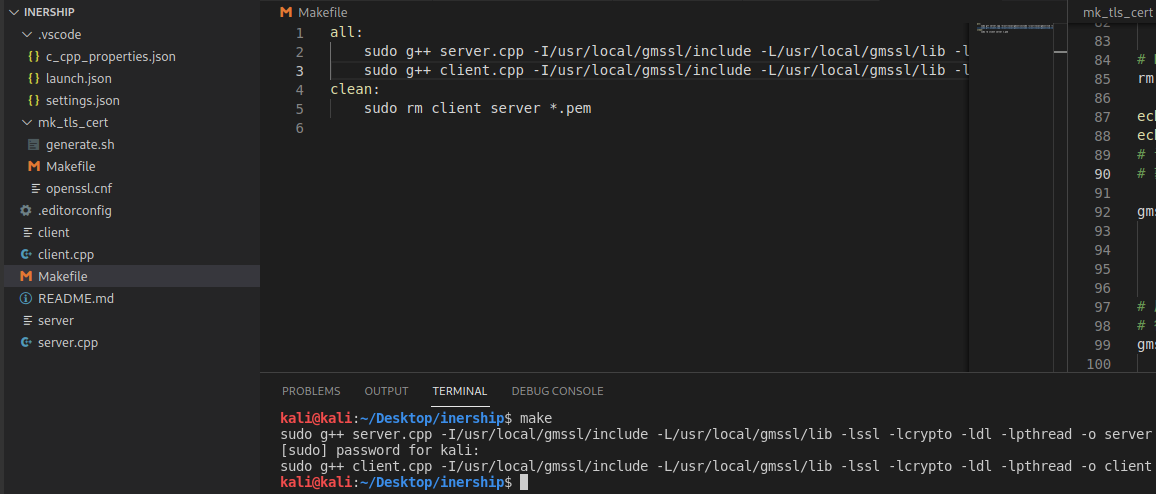
# /CN= Common Name sxy

### 2.4.7 测试

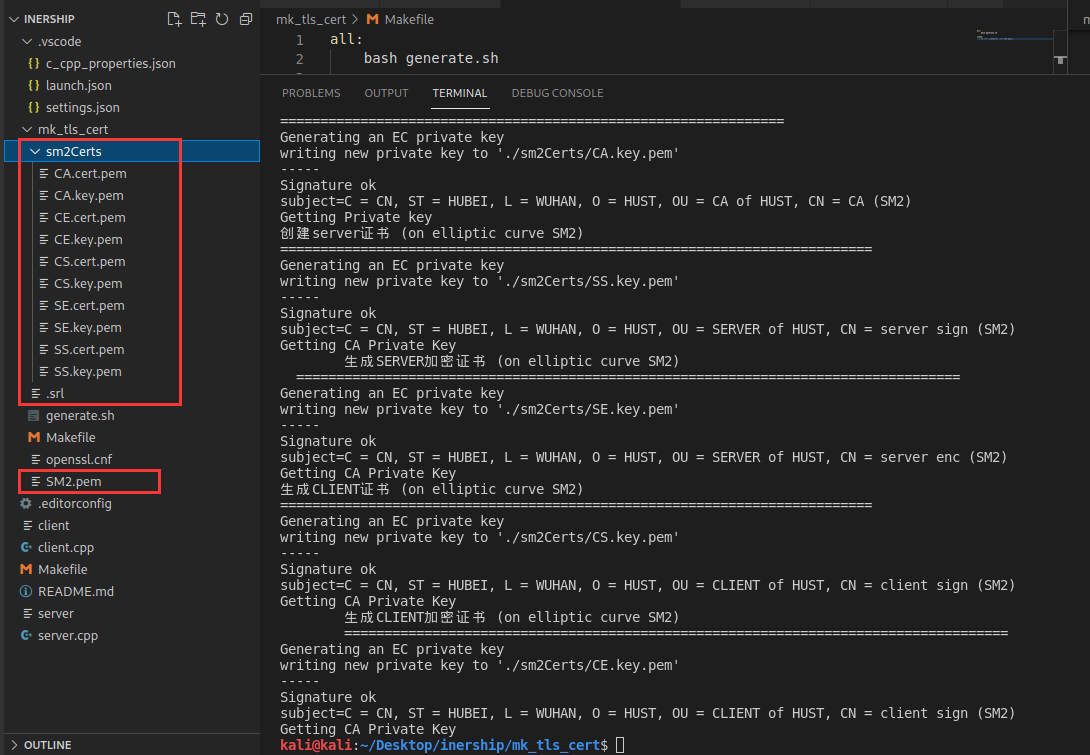
**环境：kali-linux-2020.1**

根据已经写好的Makefile文件，我们就可以简单的复现整个过程

首先编译server和client程序



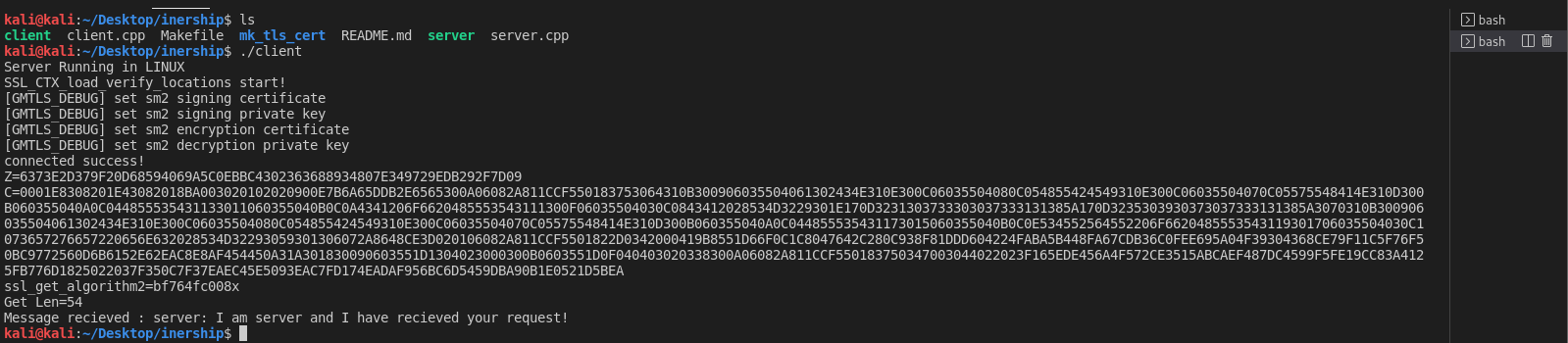
之后进入下级目录，同样使用makefile获得CA的证书与私钥、服务器的证书与私钥、服务器加密的证书和私钥、客户端的证书和私钥、客户端加密的证书和私钥



在一个terminal中执行server程序，显示在linux系统中执行，这里我用的是kali-linux



之后再开一个terminal执行client，可以看到”connected success!”，收到54个字符并显示字符串



回到server的terminal，同样可以看到类似情景，说明测试成功

## **实习工作总结**

### 2.4.1 专业知识的应用

在这次实习中，专业里学习的openssl和C语言编程给我了极大的帮助。

首先，环境配置往往是万事开头的“难”，这次也不例外。不过好在由于有专业课中老师给的部分教程，我能够跳过对于gmssl的配置的困难，直接进入后面的配置中。这一点节省了我许多时间，也给了我更多的信心去完成这次实习。

C语言编程也给了我极大的帮助。C语言是最为基础的一门计算机语言，不过因为它的基础，也使得它对于编程者并不是那么友好，集成的环境也不是那么丰富，因而许多环境需要自己设置，头文件也需要自己查找和引入，还有其他的库函数，也需要自己查询。这就得益于我们平时专业课中自己动手做过的很多实验都是基于C语言编程的，我有了丰富的解决此类问题的经验。还有socket编程也是基于C语言，这一点我们还专门在计网课中自己编写过相关程序，这样我对client和server有了更好的理解，心中早有了框架，也做得更为得心应手一些。

### 2.4.2 实习心得体会

网络安全在如今的时代越来越体现其重要性，也是因此，我参加了这次实习，主要做数据库的认证插件部分。

在这次实习中，我首先深度认识了数据库系统，了解数据库在当今社会的重要性，以及这个概念的功能性之强大，因而数据库的安全需要得到全方面的保证。正常来说，我们绝大多数数据库都是通过账号密码登录即可，然而我们已经忽略了在这个操作之前、客户端和服务器是否正常的问题，这是需要我们慎重考虑的。

这也就是我们做这个插件的原因与初衷--解决认证客户端与服务器之间的合法性的问题。

在解决这个问题之前，我们首要想到的就是，究竟用什么加密方法来对认证的信息进行加密。这是个很简单的问题，但是也是个很严重的问题。在我们上过的许多专业课上，老师都给我们讲过诸如陷门、发展之类的问题，让我印象深刻。我们专业的名称就是网络安全，所以我们理应对安全上的问题更加关注。

在过去的学习过程中，我越来越觉得AES对称加密算法如此之成熟的今天，似乎我们只需要套用开源的AES算法，事情就结束了。

在查询资料后我渐渐得之，一个公开、开源的算法并不意味着绝对的安全，算法公开并不意味着一定没有后门。可能单从程序的角度很难理解，但后门并不一定是一句看起来完全不同的代码。密码的后门可能是算法本身，也就是单看加密过程或者执行代码，不做安全分析，不可能意识到它的问题的。类比神经网络，尽管程序员可以看到训练出来的整个网络，如果不做详尽的分析，很可能无法理解为什么该网络可以解决某个问题。

不仅如此，尽管AES是欧洲学者的设计，对各国来说，依然会有不可控因素。比如大家对NSA的能力有所忌惮，美国政策是否有变，对本国数据的保护，对国外密码设备进入的限制等等。在政策层面要考虑的问题很多，发布自己的算法在世界上也实在不是什么稀奇的事。因此，既然我们要发展自己的网络安全专业，由自己从头设计一个属于自己的“国密算法”也是理所当然，这让我们尽早摆脱过多依赖于国外的技术困境！

通过研究、学习国密算法，我体会到了中国网络安全的建设者们热切而又严谨的治学心态，为中华信息安全之崛起而建立数据我们自己的算法，尽可能的摆脱过去落后的日子里对于国外技术的依赖。不仅仅是芯片，网安也是如此！我们国家在许许多多的高端领域已经看齐甚至赶超、超过欧美大国，信息安全的重要性愈加中心化，我们想要保住自己的研究成果，保住自己在那些领域的话语权，保住这些作为经济建设、社会建设、国防建设的重要“基石”，我们必须保证自己的信息传递、保存的方法是尽可能的安全的，而不是将自己的性命寄托在别人的仁慈与怜悯纸上，失去自己的原则！

接下来便是在这次实习过程中我需要使用的工具，gmssl，openssl的一个分支，主要就是集成国密算法的一个工具箱。

由于在这次实习任务中，要求必须使用C语言编程，所以也就需要了解gmssl的C语言函数依赖，这里基本用的是openssl的原套函数，这些函数都可在openssl的文档中找到。相对来说还是必须较容易的，因为openssl我在专业课上已经认证学习过，所以整个编程过程还是比较容易的，一些些常见的小问题虽然也会耗费一些时间，但分散到三周来看也就不是那么艰难。

总的来说，这次实习的经历给了我很大的启发，不仅仅是动手自己第一次参与项目工作，更是感受信息安全工作者们的拳拳报国之心，也只有因此他们才会不因openssl的便捷而忘记居安思危，将国家的未来放在心上！我们做安全行业的，首先的，不是要有过硬的技术，而是先学会做人，不然再多再强的技术都是社会的危害！

### 2.4.3 意见、不足及展望

本次实习安排的初衷其实是非常好的，让大家更早的接触未来所要从事的行业，减少踏出社会时那种慌乱与不知所措的感觉。然而，这个时候有更多的人准备考研、出国之类的方向，而实习往往是要花费大量时间去认真做成一件事，这个时间点两件重要的事情撞在一起，就显得不太合适了。个人建议提早一年实习，把这个时间错开，也让大家都有机会走出校园去实习而不是很多人为了准备考研而不得不待在校园里，没有达成最初想的那种在社会中洗涤一遍的那种效果。

我就十分遗憾，因为我的实习只有技术，而没有到大公司去体验踏出社会后的工作环境的机会，这一点，我希望以后能够找出更好的办法去调节。

# **实习问题解答**

**3.1 能认识到计算机和网络空间安全技术日新月异的发展特点，认同自主学习和终身学习的必要性，具有自主学习和终身学习的意识，结合实习的所见所闻和技术发展，简要谈谈终生学习必要性**

终生学习，即“活到老学到老”。这是一句耳熟能详的名言，从小到大，这句话在我耳畔时时响起。曾今，我曾粗略的觉得，不就是跟上时代，时代出现什么新的东西，我去了解一下即可。然而，这次实习的过程中，我深刻感受到了时代发展的迅速，一点都不等青丝成雪，也许，我们会越来越没法跟上时代的浪潮，如果我们仅仅是去了解一下。

光是计算机语言就产生着非常大的变化。

虽然实习安排中写着只能用C语言编程完成任务，但我首先想到的就是go语言的快捷与友好的环境。Go（又称Golang）是[Google](https://baike.baidu.com/item/Google" \t "https://baike.baidu.com/item/go/_blank)开发的一种[静态](https://baike.baidu.com/item/%E9%9D%99%E6%80%81" \t "https://baike.baidu.com/item/go/_blank)[强类型](https://baike.baidu.com/item/%E5%BC%BA%E7%B1%BB%E5%9E%8B" \t "https://baike.baidu.com/item/go/_blank)、编译型、并发型，并具有垃圾回收功能的[编程语言](https://baike.baidu.com/item/%E7%BC%96%E7%A8%8B%E8%AF%AD%E8%A8%80" \t "https://baike.baidu.com/item/go/_blank)。

Go于2009年11月正式宣布推出，成为[开放源代码](https://baike.baidu.com/item/%E5%BC%80%E6%94%BE%E6%BA%90%E4%BB%A3%E7%A0%81/114160" \t "https://baike.baidu.com/item/go/_blank)项目，并在[Linux](https://baike.baidu.com/item/Linux" \t "https://baike.baidu.com/item/go/_blank)及[Mac OS X](https://baike.baidu.com/item/Mac OS X" \t "https://baike.baidu.com/item/go/_blank)平台上进行了实现，后来追加了Windows系统下的实现。在2016年，Go被软件评价公司TIOBE 选为“TIOBE 2016 年最佳语言”。目前，Go每半年发布一个二级版本。

与C++相比，Go并不包括如[枚举](https://baike.baidu.com/item/%E6%9E%9A%E4%B8%BE" \t "https://baike.baidu.com/item/go/_blank)、[异常处理](https://baike.baidu.com/item/%E5%BC%82%E5%B8%B8%E5%A4%84%E7%90%86" \t "https://baike.baidu.com/item/go/_blank)、[继承](https://baike.baidu.com/item/%E7%BB%A7%E6%89%BF" \t "https://baike.baidu.com/item/go/_blank)、[泛型](https://baike.baidu.com/item/%E6%B3%9B%E5%9E%8B" \t "https://baike.baidu.com/item/go/_blank)、[断言](https://baike.baidu.com/item/%E6%96%AD%E8%A8%80" \t "https://baike.baidu.com/item/go/_blank)、[虚函数](https://baike.baidu.com/item/%E8%99%9A%E5%87%BD%E6%95%B0" \t "https://baike.baidu.com/item/go/_blank)等功能，但增加了 切片(Slice) 型、并发、管道、垃圾回收、接口（Interface）等特性的语言级支持。

就我的实习任务来说，go语言有丰富的标准库，继承了非常多的功能，很多用C语言需要几十行代码才能完成的任务，用go语言只需要一个函数应用即可。特别是有关openssl的函数，这里设计的更为简单、明了，如果能使用golang也许不会再踩那么多的坑。

然而，我这里说golang其实是为了说明语言发展的一个方向。计算机编程语言正在越来越快的朝着集成化的方向发展，对比C语言和golang就可以知道，类似的语言还有python、java等。也许过不了多久，当我还在用这些基础的语言费很大劲才能编写一个软件时，使用新语言的编程者只需要几行代码就可以完成。

同理，安全行业也是这样，越来越多的工具软件正在被开发或准备开发，越来越多的新技术也在被引进，老人难以想象新人的便利，如果，不真正参与进去，陪伴技术的发展。

Gmssl是openssl的一个分支，但与原来的openssl相比，独立出来的gmssl在许多地方已经发展的更远了，如果我们还在使用老旧的版本，也许我们将永远握法理解愈发年轻的世界。

这给我很大的启示，我将终身自主学习我投身的行业，而非作为一个老人守旧而顽固，这是不正确的。不过我想我一定会是这样，我喜欢新的事物，喜欢新的尝试，也喜欢新的体验。去尝试、去感受、去参与，也许就是我这次实习过程中最大的收获了吧。

|  |
| --- |
| **实习导师评语及评分 分数：** |
| 签 字：  时 间： |