物联网网关轻量级认证和加密技术研究

【原文对照报告-大学生版】

报告编号：f92f3e2fe966307d　　检测时间：2020-05-27 11:11:58　　检测字数：18,708字

作者名称：郑智聪　　所属单位：维普论文检测（河北）

检测范围：

◎ 中文科技期刊论文全文数据库
◎ 中文主要报纸全文数据库
◎ 中国专利特色数据库
◎ 博士/硕士学位论文全文数据库
◎ 中国主要会议论文特色数据库
◎ 港澳台文献资源
◎ 外文特色文献数据全库
◎ 维普优先出版论文全文数据库
◎ 互联网数据资源/互联网文档资源
◎ 高校自建资源库
◎ 图书资源
◎ 古籍文献资源
◎ 个人自建资源库
◎ 年鉴资源
◎ IPUB原创作品

时间范围：1989-01-01至2020-05-27

检测结论：

全文总相似比		复写率		他引率		自引率		专业术语
14.62%	=	6.72%	+	7.9%	+	0.0%	+	0.0%

其他指标：

自写率：85.38%

专业术语：0.0%

高频词：密钥,网关,加密,服务,服务器

典型相似性： 无

指标说明：

复写率：相似或疑似重复内容占全文的比重

他引率：引用他人的部分占全文的比重，请正确标注引用

自引率：引用自己已发表部分占全文的比重，请正确标注引用

自写率：原创内容占全文的比重

专业术语：公式定理、法律条文、行业用语等占全文的比重

典型相似性：相似或疑似重复内容占互联网资源库的比重，超过30%可以访问

总相似片段：102

期刊：22 博硕：42 外文：0 综合：0 自建库：10 互联网：28

维普论文检测（河北）

VPCS 维普论文检测系统
VIP PAPER CHECK SYSTEM

VPCS 维普论文检测系统 原文对照

颜色标注说明：

■ 自写片段

■ 复写片段（相似或疑似重复）

■ 引用片段

■ 专业术语（公式定理、法律条文、行业用语等）

河北北方学院
HEBEI NORTH UNIVERSITY

学士学位论文

物联网网关轻量级认证和加密技术研究

姓 名 郑智聪

学 号 201642030

院 系 信息科学与工程学院

专 业 电子信息科学与技术

指导教师 郭本振

二〇二〇 年 六 月 一 日

物联网网关轻量级认证和加密技术研究

Lightweight authentication and encryption technology for IoT gateway

Technical research

学位论文原创性声明

本人所提交的学位论文《物联网网关轻量级认证和加密技术研究》，是在导师的指导下，独立进行研究工作所取得的原创性成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出重要贡献的个人和集体，均已在文中标明。

本声明的法律后果由本人承担。

论文作者（签名）： 指导教师确认（签名）：

年 月 日 年 月 日

学位论文版权使用授权书

本学位论文作者完全了解河北北方学院有权保留并向国家有关部门或机构送交学位论文的复印件和磁盘，允许论文被查阅和借阅。本

VPCS 维普论文检测系统
VIP PAPER CHECK SYSTEM

人授权河北北方学院可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或其它复制手段保存、汇编学位论文。

保密的学位论文在_____年解密后适用本授权书。

论文作者（签名）： 指导教师（签名）：

年 月 日 年 月 日

摘 要

现如今，物联网越来越广泛地应用在人们的生活中，其快速发展的同时，物联网的安全问题也日益突出。在基于物联网技术设计的温室环境监控系统中，需要实现云服务器与物联网节点设备（如传感器、控制器等）之间的双向身份认证和数据加密方案。本课题研究了基于椭圆曲线ECC的数字签名(ECDSA)和密钥协商(ECDH)算法，配合AES-128算法加密传输数据，验证了上述算法在实际项目中的可行性，能够保证物联网系统的安全。通过实际调试和实验证明该方案可在资源较低的情况下，满足云服务器与物联网网关设备之间进行双向身份认证的设计需求，对本方案所实现的功能进行了全面的测试，系统运行稳定，操作简单。

关键词：物联网安全；双向认证；密钥协商；椭圆曲线；安全策略

ABSTRACT

Nowadays, the Internet of things is more and more widely used in people's life, but along with its rapid development, its security problem has become increasingly prominent. In the greenhouse environment monitoring system designed based on Internet of things technology, it is necessary to realize bidirectional identity authentication and data encryption scheme between cloud server and Internet of things node devices (such as sensors, controllers, etc.). In this project, Elliptic Curve Digital Signature Algorithm (ECDSA) and ECDH algorithms based on elliptic curve crypto (ECC) are studied. The data transmission is encrypted with AES-128 algorithm, and the feasibility of the above algorithm in practical projects is verified, which can ensure the security of the Internet of things system. Through actual debugging and experiments, it is proved that this project can meet the design requirements of bidirectional identity authentication between cloud server and Internet of things gateway device in the case of low resources. After a comprehensive test of the functions of this scheme, the system is proved to be stable and easy to operate.

Key Words: IoT security; Two-way authentication; Key agreement; Elliptic curve; Security strategy

目 录

摘 要I

目 录III

第1章 引言1

1.1 研究背景1

1.2 国内外研究现状1

1.2.1 国外研究现状1

1.2.2 国内研究现状2

1.3 主要研究内容2

1.4 研究意义2

第2章 相关理论和技术4

2.1 对称和非对称加密4

2.1.1 对称加密4

2.1.2 非对称加密5

2.2 数字签名技术6
2.3 椭圆曲线ECC6
2.3.1 概述6
2.3.2 ECC上的运算7
2.3.3 ECC上的离散数对问题9
2.3.4 椭圆曲线上的DH密钥交换（ECDH）9
2.4 AES加密算法10
第3章 需求分析14
3.1 可行性分析14
3.1.1 技术可行性分析14
3.1.2 经济可行性分析14
3.1.3 操作可行性分析14
3.2 农业物联网安全体系及需求14
第4章 基于ECC、AES的轻量级认证加密算法16
4.1 ECC算法的性能瓶颈分析16
4.2 ECC算法的改进16
4.3 基于改进ECC-AES的混合加密方案设计17
4.3.1 网关认证与密钥协商17
4.3.2 数字签名与数据加密18
4.3.3 密钥更新20
4.4 算法安全性分析21
第5章 系统测试23
5.1 测试环境23
5.2 系统功能测试23
5.2.1 密钥协商测试24
5.2.2 数字签名和加密解密测试25
5.2.3 心跳管理测试26
5.2.4 密钥更新测试27
5.3 系统性能测试28
5.3.1 并发压力测试28
5.3.2 安全性测试28
第六章 结论30
谢辞31
参考文献32
附录33

第0章

第1章引言

本章简要说明了本课题的研究背景，物联网安全问题的国内外研究现状以及物联网网关轻量级认证和加密技术的研究意义。

1.1 研究背景

维普论文检测（河北）

1999年，MITAuto-ID中心的Ashton 教授最早提出了物联网这个名字，2005 年在国际电信联盟（ITU）发布的《ITU 互联网报告 2005：物联网》报告中，再次提出用了“物联网”的概念[1]。

在物联网蓬勃发展的同时，出现了很多针对物联网的攻击事件，人们越来越关注物联网系统的安全问题。2015年黑客针对乌克兰的电力系统发起恶意攻击，导致70多万居民家庭停电数小时；2016年的Mirai事件中，攻击者利用网络摄像头等大量的物联网设备向域名服务器发起DDoS攻击，导致大量用户无法使用网络。据Gartner调查，全球近20%的单位和部门，近年来遭受过物联网攻击。

一般的物联网体系主要由三层组成，从上往下分别为：应用层、传输层、

感知层。物联网一般的工作模式是：感知层负责感知周围的信息，并通过传输层连接上应用中心，应用中心负责数据的汇集、分析等。其中传输层和应用层可以在现有的、成熟的架构基础上运作实施，这两层的安全保护都有成熟的认证体系，

而对于感知层，因为硬件资源受限、传感器节点分布较广泛等特点，现有的安全技术无法很好地实施，迫切需要一种轻量级的认证和加密体系。

1.2 国内外研究现状

1.2.1 国外研究现状

近年来，随着物联网中的安全问题日益暴露，世界各地越来越多的官方组织、学术机构加入到物联网安全的研究当中，旨在构建一系列的安全规范和协议。

Nicanfar 等[2]在 2015 年提出了一个基于椭圆曲线加密的认证协议。虽然他们的方案满足了降低计算复杂度的需求，但是第三方仍然需要用密码表来保存用户信息，在表丢失或被窃取的情况下不能保证信息安全[18]。之后，Li等[3]提出一个新的密码协议但同样也被证明易受到窃听，并且由于缺少密钥协商而易受到模仿攻击。2019年Q.Jiangetal.[4]在分析Das协议的基础上，提出了一种新的在云服务器协助下，实现可穿戴设备认证和密钥协商的安全协议。该协议使用了ECC算法进一步增强了数据的安全性，降低了协议对设备计算资源消耗。2019年Wangetal[5] 提出了一种使用ECC算法和云服务器辅助的物联网网关与用户智能手机进行双向身份认证和密钥协商的协议。

1.2.2 国内研究现状

2017年，汪洋[1]提出了一种新的 RFID 双向认证协议，在认证信息中加入随机数和时间戳，利用椭圆曲线密码算法（Elliptic curve cryptography，ECC）对认证过程中的敏感信息进行加密，保证认证信息的机密性。2018年，史冰清[7]设计实现了能够在物联网网关上使用的混沌-AES加密算法，该算法密匙混沌化、密钥空间更大、实现了“一块一密”，并且没有增加密钥管理的负担。同年，王斌[8]在针对物联网不同的层次结构，针对性地设计了应用于不同层次结构的安全防护策略。

1.3 主要研究内容

本文针对当前物联网系统感知层设备多数使用对称密钥协议来构建网络安全基础设施的实际场景，考虑到密钥维护工作繁多，运维人员经常接触通信密钥，容易引起密钥泄漏和内部特权人员攻击的情况。以及感知层设备计算、存储和通信资源有限，攻击者容易发起资源耗尽型的DDos攻击的情况。使用ECC加密算法设计了一种系统云认证服务器（Cloud Authentication Server CAS）与物联网网关设备之间的双向认证、密钥协商的解决方案。本课题主要研究内容有以下几点：

1.物联网网关与服务器之间密钥协商，双方在不暴露私钥的情况下计算出相同的加密密钥。

2.数据加密，通信双方认证通过后，发送的数据都是通过加密的密文。

3.数字签名与验签，通信双方发送数据时进行数字签名，接收数据时进行验签，确保消息没有被修改。

4.密钥管理，协商出的加密密钥可以人为进行更新，而不是一成不变。

1.4 研究意义

对万事万物的感知能力是物联网技术的精髓，因此，感知层构成了物联网体系中举足轻重的一部分。感知层通过传感器等设备，借助于蓝牙、无线网络等传输信息到物联网中心系统，其结构图如下：

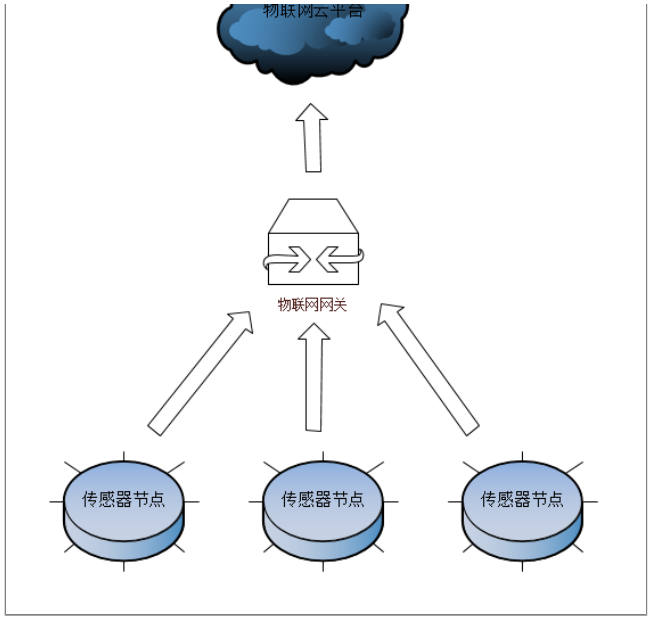


图 1-1 感知层结构图

在一般的物联网系统中，通常使用AES,DES等对称加密算法，但是对称加密算法又存在着密钥泄露的风险，一旦攻击者通过非法手段获取了密钥，后果不堪设想。而嵌入式设备又有着受限的资源，严苛的工作环境等特点，使用RSA等非对称加密算法的话，大大影响传感器的工作效率。因此，迫切的需要一种适用于嵌入式设备的轻量级认证和加密协议，在保障安全性的基础上，尽量降低带宽、内存等资源的占用，提高密钥协商和加密解密的效率，这对于物联网安全的发展有着深远的意义。

第2章相关理论和技术

本章介绍了课题研究用到的信息加密相关理论，详细描述了数字签名、椭圆曲线、AES加密算法的原理及相关步骤。

2.1 对称和非对称加密

人们的信息在网络中传输，很容易被不法分子拦截并篡改，轻则造成人们隐私数据的泄露，重则造成人民个人财产的损失。而对网络信息进行加密则是保证机密信息和数据泄露的主要手段，以下为密码学中的一些基本概念。

（1）加密：将数据按照一定的规则进行变换的过程

（2）解密：将加密后的数据，按照规则转换成原数据的过程

（3）明文：加密之前的数据，能够被轻易读取

（4）密文：加密之后的数据，隐藏原文本的信息

（5）密钥：控制加密和解密过程的参数

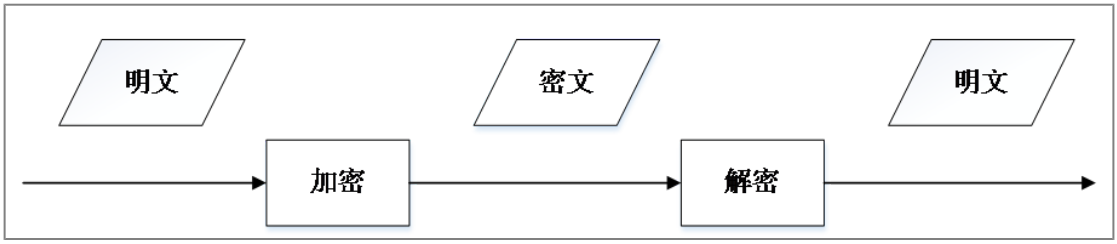


图 2-1 数据加密解密过程

2.1.1 对称加密

对称加密的核心原理是对数据进行加密和解密时密钥是一致的，特点是加密解密的速度非常快，实施起来较为简单的同时也能有较好的安全性。对称加密的过程如下图所示：

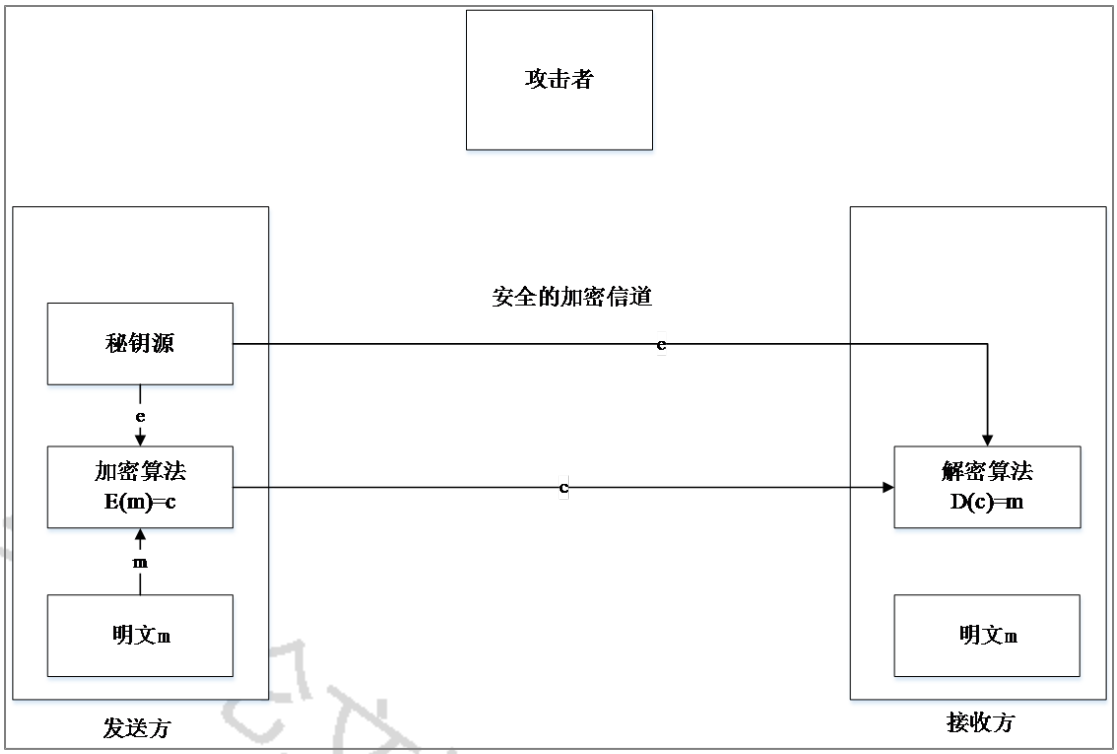


图 2-2 对称加密模型

要想保证对称加密算法保障的安全性，必须考虑如何保证密钥的安全，通信的双方都不能将密钥泄露出去，否则会面临密码被破解的危险。而在物联网的环境中，网关节点等距离服务器都比较远，所以密钥在网络中传送是不可避免的，而且很容易在密钥传送过程中被拦截并篡改。

2.1.2 非对称加密

非对称加密算法最早产生于上世纪七十年代[9]，与对称加密不同的是，非对称加密需要公钥和私钥来进行加密和解密，公钥和私钥总是成对出现，只有持有相应的私钥才能对公钥加密的数据进行解密。其中公钥可以发布到网络上，任何人都可以获得公钥，不存在密钥泄露的问题。

1. RSA算法

RSA密码体制是由 Rivest、Shamir 和 Adleman 在 1977 年联合提出的[1]。RSA 算法的原理是基于一个数论事实：将两个大素数相乘很容易，但是想要对其乘积进行因式分解却极其困难，因此可以将乘积公开作为公钥[12]。

2.ECC算法

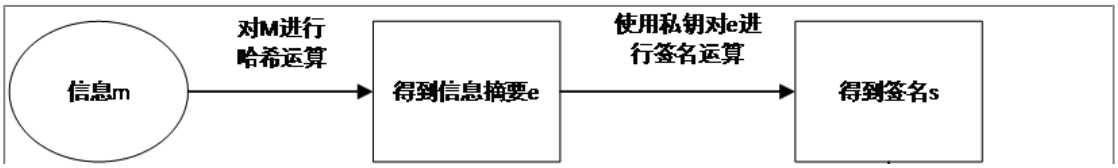
ECC算法是在1985年由Neal Koblitz和Victor Miller分别独立提出的[13]。

相比于RSA算法，ECC可以做到在同等的安全强度下，具有更小的密钥长度。ECC算法的原理是定义椭圆曲线上的标量乘运算，然后利用离散对数难题，将标量乘运算的结果作为公钥。本文将在2.3小节详细介绍ECC算法。

2.2 数字签名技术

在信息安全领域，数字签名是公钥加密体制的重要应用，就像在实际世界上用手写的签名和印章一样，通过在原始数据上附加一些签名数据,或是对原始数据进行编码作为数字签名。使用数字签名可以确保数据在网络中传输的过程中被伪造或修改。

数字签名技术能够验证信息的完整性，其流程如下：



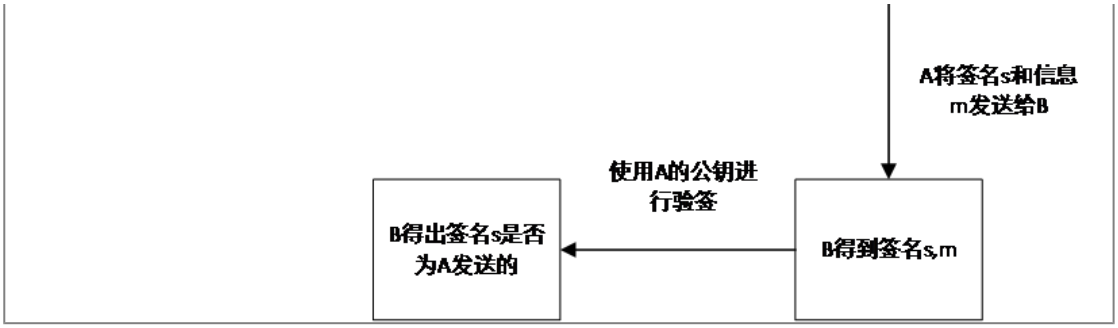


图 2-3 数字签名流程

常见的数字签名算法有RSA、DSA等[14]，本课题的基于椭圆曲线的数字签名算法是在DSA的基础上，利用椭圆曲线的离散对数难题，提高了安全性和计算效率。

2.3 椭圆曲线ECC

2.3.1 概述

本文研究的椭圆曲线是指满足式2-1的方程

$y^2 = x^3 + ax = b \bmod p where(x, y) \in Z_p$	式（2-1）

式2-1中p是一个较大的素数，且p值越大安全性越高，计算量也就越大。a，b∈ Ep用于确定具体的椭圆曲线，且需要满足公式2-2:

$4a^3 + 27b^2 \neq 0$	式（2-2）

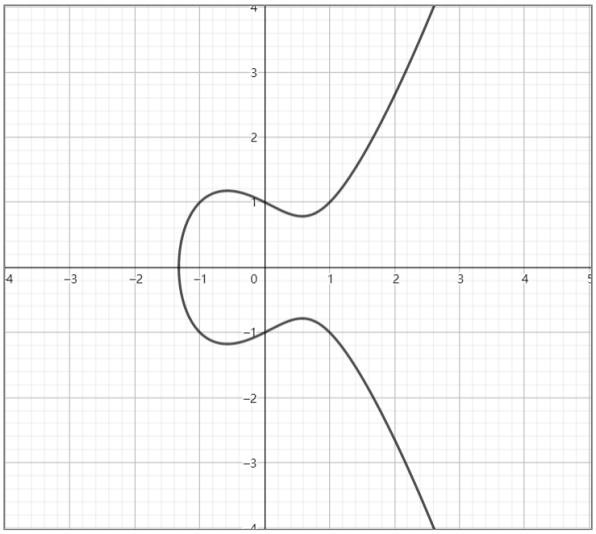


图 2-4 椭圆曲线y^2 =x^3-x+1

2.3.2 ECC上的运算

维普论文检测（河北）

1.ECC上的加法

在椭圆曲线上取两点A和B做直线AB，AB过点C与椭圆曲线相交，关于X轴做C的对称点D，则D就是A、B两点的和。

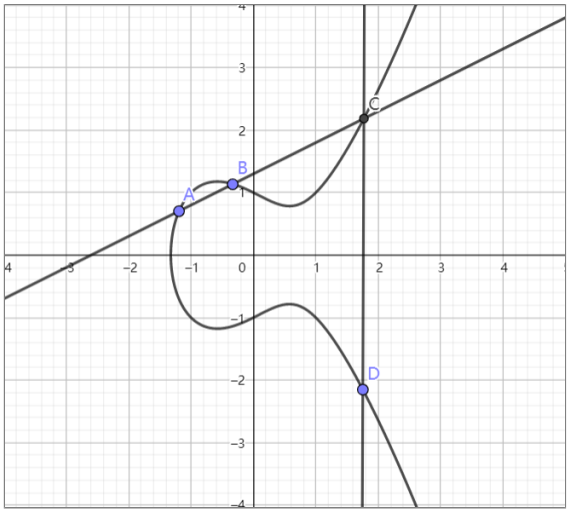


图 2-5 椭圆曲线加法

在一种特殊情况下A、B点重合时，过AB点做切线，与椭圆曲线相交于R点，关于X轴做R的对称点R’，则R’就是A、B两点的和。

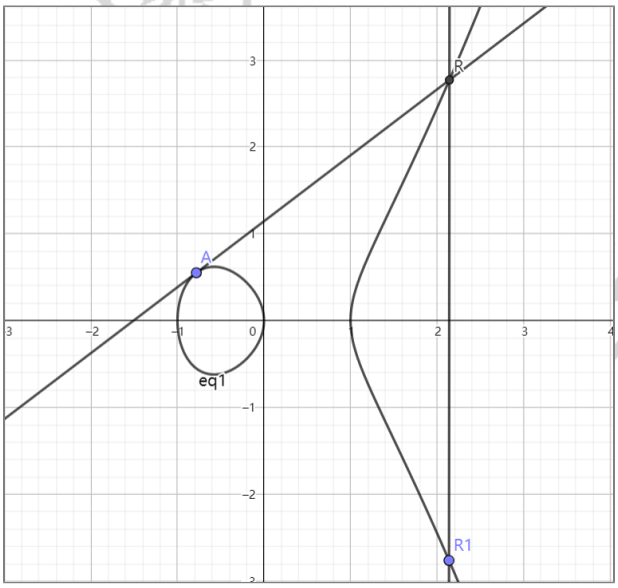


图 2-6 A=B时椭圆曲线加法

另一种特殊情况下点A、B所在的直线刚好平行于Y轴时，根据椭圆曲线关于X轴对称的性质，将永远不会有与曲线的第三个交点。定义一个无穷远的特殊点，称为0点（零点），那么直线AB相交椭圆曲线与0点，记作A+B=0。

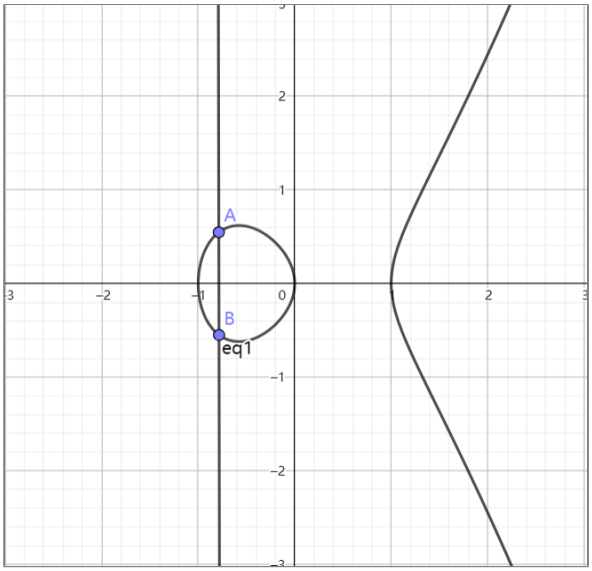


图 2-7 AB垂直于X轴时椭圆曲线加法

2.ECC上的乘法

给定椭圆曲线上的一个点P，计算kP就相当于计算k个P相加，这种运算过程称为标量乘运算。由于椭圆曲线上的加法满足结合律[11]，那么

P+P+P+P = 2P+2P

这样一来，假设要计算16P，则可以先计算出来2P，然后计算4P，如此类推，可以把16次的加法运算减少到4次，这个算法的时间复杂度是O(logk)，相比于直接进行倍点运算降低了很多，对于增加ECC算法的效率有十分重要的影响。

2.3.3 ECC上的离散数对问题

当给定基点G时，可以很容易的根据ECC乘法运算规则计算出xG，但是如果已知xG，要求x却是几乎不可能的事情，这就是椭圆曲线密码体制中所利用的“椭圆曲线上的离散对数问题”。

2.3.4 ECC上的DH密钥交换（ECDH）

ECC上的DH密钥交换（ECDH）是利用椭圆曲线的离散数对问题，能够实现在不安全的信道里面，使通信的双方协商出一个共同的对称加密密钥。

ECDH的流程如下

1.A选定一条椭圆曲线E基点G。

2.A选择一个随机数k作为私钥，计算公钥K=kG。

3.A将E和点K、G发送给B。

4.B选择一个随机整数r（r<n）。

5.B计算点R=rG，key = rK = rkG（key就是交换出的密钥）

6.B将R传给A

7.A收到信息后，计算key = kR = krG

数学原理:key = rK = rkG = kR = krG

至此：A和B就协商出来相同的密钥，后面就可以使用这个密钥进行对称加密通信。攻击者只能获取中间在信道中传输的E和点K、G、R等，由于椭圆曲线的数学难题，无法计算出A和B的密钥k、r，那么协商出来的密钥key也就是安全的。

维普论文检测（河北）

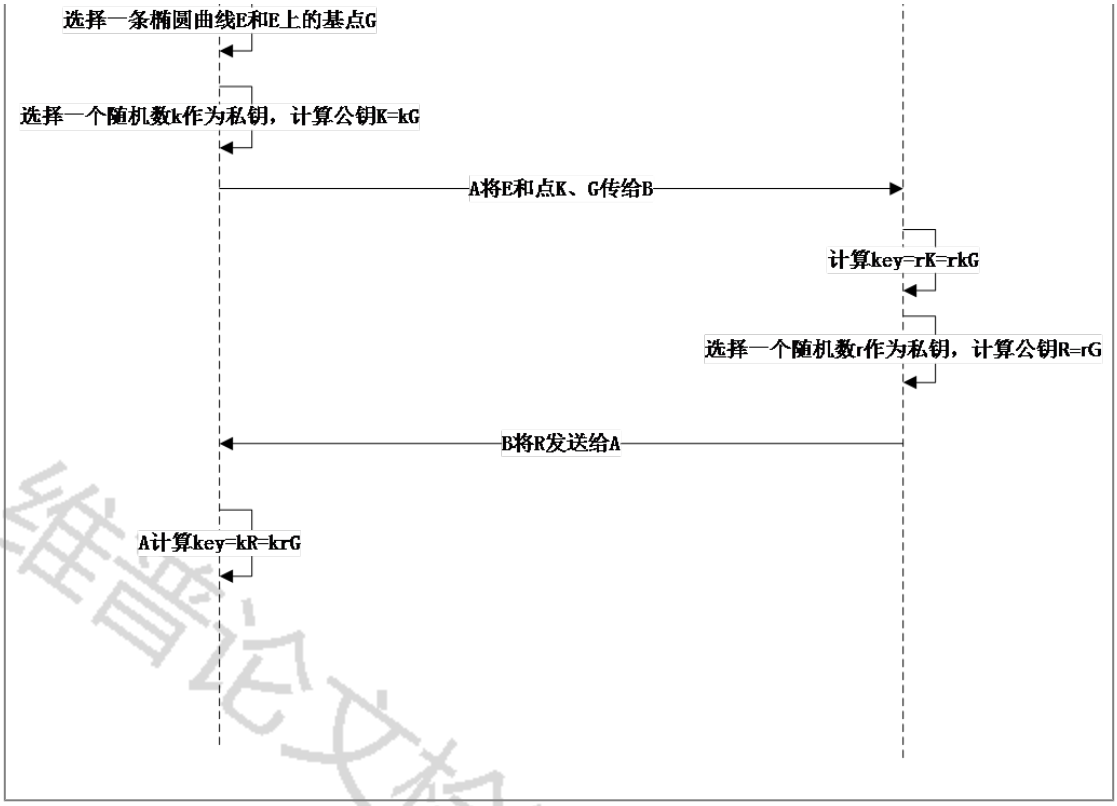


图 2-8 ECDH密钥交换

2.4 AES加密算法

AES算法最初是由比利时密码学专家Joan Daeman和Vincent Rijmen设计和提出，又叫做Rijmen算法[10]。根据密钥的长度不同可以分为AES128、AES192、AES256三种，本文使用的是AES128，每一种的加密轮数如下表

表 2-1 三种AES算法加密轮数

AES	密钥长度	分组长度	加密轮数
AES128	4	4	10
AES192	6	4	12
AES256	8	4	14

AES算法的特点就是比较轻量级，有较高的计算效率，在硬件设备时实施简单，并且有的芯片有专门的AES硬件加速。AES采用轮加密的方式，加解密的关键步骤如下：

1.字节代换与字节逆代换

在AES算法中，设计了两个二维矩阵，分别称为S盒和逆S盒，在进行字节代换操作时，通过查表的方式把明文字节按照矩阵进行转换，如下图所示

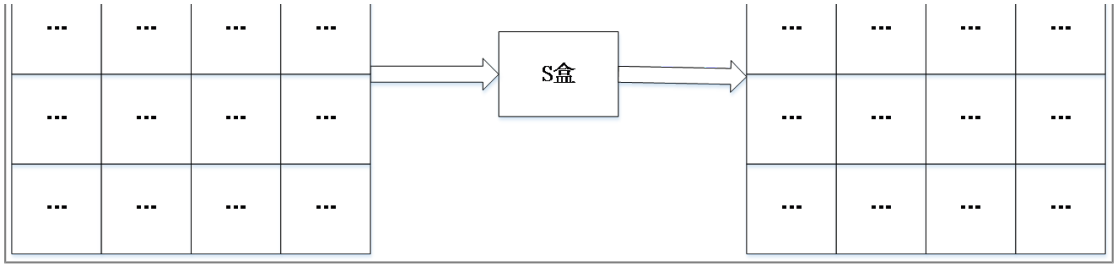


图 2-9 AES字节代换

2.行移位变换和行移位逆变换

行移位变换就是循环向左进行移位。在AES128中，状态矩阵向左移的字节数是依据行数递增的，如下图所示：

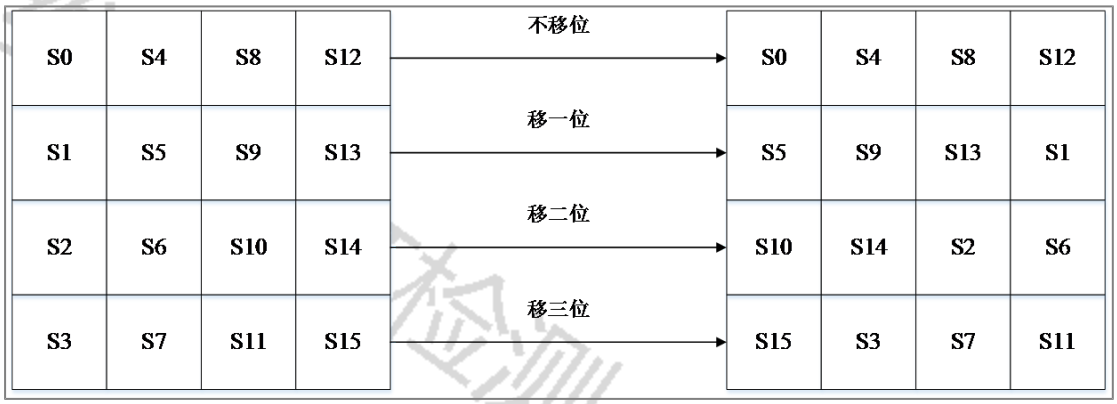


图2-10 AES行移位

3.列混合与列混合逆运算

列变换就是通过式2-3进行矩阵相乘，等号右边分别为行移位后的状态矩阵和一个常矩阵。

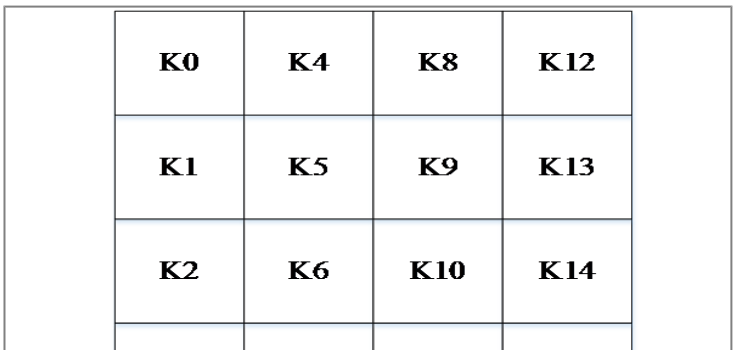
$$\begin{bmatrix} S'_{00} & S'_{01} & S'_{02} & S'_{03} \\ S'_{10} & S'_{11} & S'_{12} & S'_{13} \\ S'_{20} & S'_{21} & S'_{22} & S'_{23} \\ S'_{30} & S'_{31} & S'_{32} & S'_{33} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{00} & S_{01} & S_{02} & S_{03} \\ S_{10} & S_{11} & S_{12} & S_{13} \\ S_{20} & S_{21} & S_{22} & S_{23} \\ S_{30} & S_{31} & S_{32} & S_{33} \end{bmatrix}$$

4.轮密钥加

轮密钥是将初始密钥通过轮密钥产生算法运算出来的，轮密钥加就是将状态矩阵中的数据的每一位与轮密钥进行异或。

5.密钥扩展

密钥扩展是轮密钥加中提到的轮密钥产生算法。首先用一个4*4的状态矩阵来盛放初始密钥，如下图所示



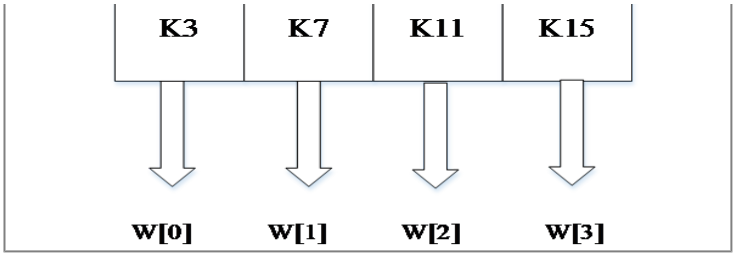


图2-11 初始密钥状态矩阵

矩阵的每一列的4个字节组成一个字，4个字依次命名为W[0]、W[1]、W[2]、W[3]，构成一个以字为单位的数组W，接着对 W 数组进行四次扩充，得到一个共 44 列的密钥扩展数组[16]，密钥扩展过程如图 2-13 所示。

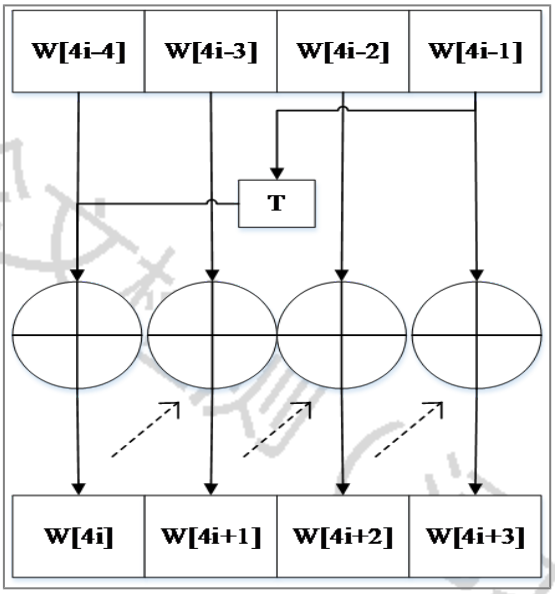
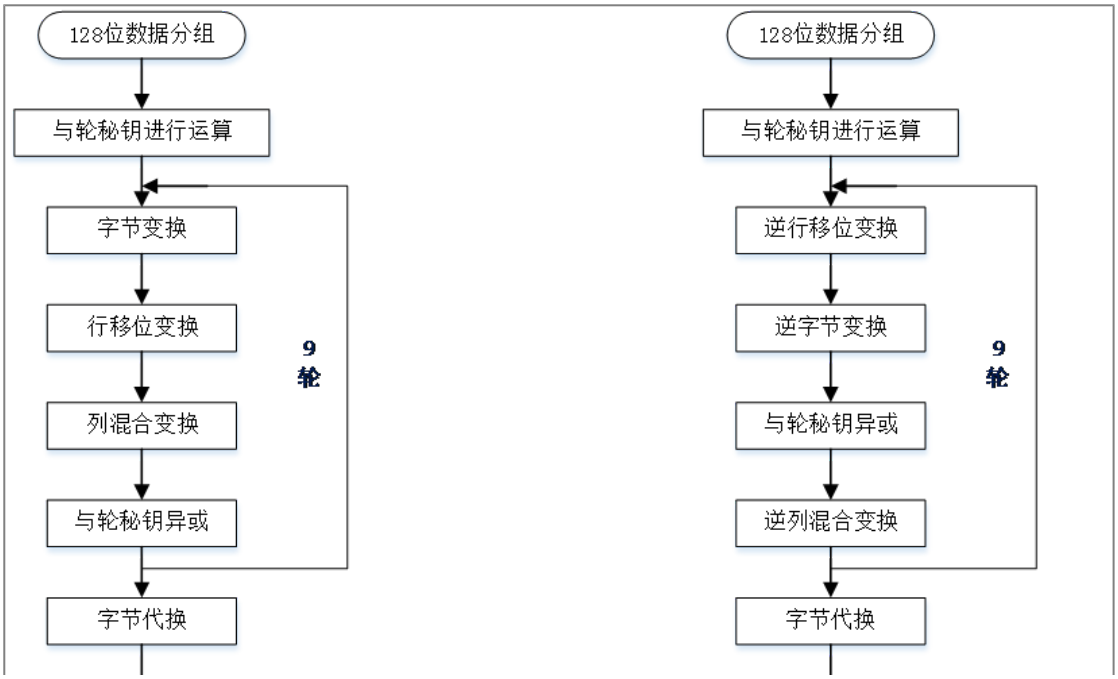


图2-12 密钥扩展过程

6.AES加密解密的流程图如下



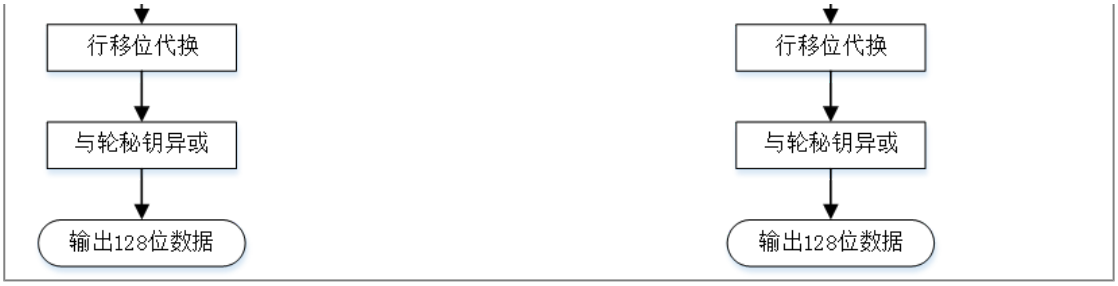


图2-13 AES加密流程图 图2-14 AES解密流程图

第3章需求分析

本章分析了系统方案的可行性，确定了项目是否可以实施，然后从农业物联网的特性出发，明确了系统需要实现的主要功能需求。

3.1 可行性分析

3.1.1 技术可行性分析

本设计利用开源的Netty框架实现网络通信，通过设计合理的数据结构来完成认证加密流程，使用ECC进行密钥协商、数字签名，使用AES进行加密解密。系统中涉及到的技术目前已经比较成熟，借助大学四年所学到的知识，以及上网查阅官方文档，请教经验丰富的老师等等方法对未知的难点进行攻克，因此在技术方面是可行的。

3.1.2 经济可行性分析

本设计的核心在于认证和加密算法。软件方面的投入主要在于研发、调研、后期维护、域名以及云服务器方面，云服务器方面可以使用阿里云的学生优惠服务器，内存大小够用并且费用较低。因此本设计在经济上是可行的。

3.1.3 操作可行性分析

从操作上来讲，农业物联网的网关可以随机部署在农场的各个地方，随机性比较大，网关上电后，可自行与云服务器进行密钥协商与双向认证，后续的数据都是加密传输。用户可在PC查看当前在线的网关列表，也可直接通过Web端直接刷新对应网关的加密密钥，操作性强。

3.2 农业物联网安全体系及需求

传统农业有着生产效率低下、人工成本高、产量较低、无法远程管理和自动化的问题，农业物联网的出现则很好的解决了这个问题。但是，目前大多数的农业物联网系统都没有在物联网的安全问题方面深入考虑，存在着安全隐患，容易出现造成严重的后果。例如：

1.传感器参数篡改，例如氨气浓度参数被篡改，导致错过报警时机。

2.设备非法启动、停止，例如风机无故启动、湿帘无故关闭。

3.传感器冒充，收到非法的监测数据，遭受DDOS攻击。

本文提出的轻量级的双向认证和加密方案中实现的主要功能如下：

5.物联网网关与服务器之间密钥协商

物联网网关上电之后，会主动向服务器发送网关认证请求，双方使用ECDH算法，协商出网关与服务器之间的数据传输密钥。

6.网关与服务器之间加密通信

网关与服务器之间的通信是全双工通信，网关向服务器上报数据时，使用

之前协商出来的加密密钥进行AES128加密。网关收到服务器数据时，进行AES128解密。

7.连接管理

服务器上维护着一个网关的连接列表，可以进行批量的数据推送，网关流

分析与监控等操作。

8.心跳管理

网关和服务器都要监听彼此之间的通信信道，当信道在一段时间内没有数据

维普论文检测（河北）

读写时，向对方发送心跳。服务器如果长时间没有接收到对方的心跳应答，会将对应的网关标记为下线，后续可以生成网关异常告警，自动生成工单并指派工程师维修等。而网关会标记服务器崩溃，并开启定时任务进行自动重连。

9.密钥刷新

用户可以在web页面上点击按钮，刷新指定网关的加密密钥。

第4章基于ECC、AES的轻量级认证加密算法

本章首先分析了ECC算法的性能瓶颈与改进方法，在此基础上介绍了本方案实现密钥协商、数字签名、数据加密、密钥管理的细节，并对本方案的安全性进行了分析。

4.1 ECC算法的性能瓶颈分析

与RSA等其他非对称加密方式相比较，在相同安全性的要求下，ECC算法的密钥更短。但是与AES、DES等对称加密算法比较，ECC算法的时间复杂度更高，大大限制ECC算法在资源较低的处理器中的实施。而在本文研究的农业物联网安全通信中，出于成本考虑，物联网网关、节点、传感器等计算资源都是十分有限，如果直接应用ECC算法，将对系统的主要功能和实时性造成较大影响。

根据前面对椭圆曲线算法原理的分析可知，在计算过程中耗时最长的是标量乘计算，要想提高椭圆曲线算法的运算效率，瓶颈在于标量乘运算的效率。

4.2 ECC算法的改进

传统的标量乘计算过程主要是包括点加运算和倍点运算。阅读文献可知，现有的改进ECC标量乘运算方法有两类：

1.将整数k使用公式进行展开，通过控制展开式中非零元素的个数，从而降低点加和倍点运算次数[1]，提高其运算效率。例如双基链表示法[1]、二进制算法[9]等。

2.以空间换时间，通过存储前面计算的结果来进行预计算。具体实现方式是存储与点G相关的计算结果，在后续的计算中通过查表的方式实现快速地计算kG，以降低点加和倍加运算的次数，提交运算效率。例如滑动窗口法及结合NAF方法的 w-NAF窗口法[1]。

4.3 基于改进ECC-AES的混合加密方案设计

4.3.1 网关认证与密钥协商

为了实现物联网网关和原服务器的双向认证和密钥协商，我们需要作出如下约定。

1.所有设备包括云认证服务器CAS，物联网网关之间共享一组公共的参数。其中包括一组EC参数E{a,b,p,G,n,h}；独立的身份ID和时钟。身份标示ID用于在系统中唯一标识指定设备，其他设备需要通过ID访问指定的设备。设备时钟用于保证消息的新鲜。

2.云服务器和网关约定好了网关登录口令，用来验证密钥的正确性。

其基本流程如下。

第一步.云服务器初始化时，选取一个随机的大素数p，通过式（4-1）计算出ECC上的点P作为服务端的公钥，而p作为服务端的私钥。

$\mathbf{P} = p\boldsymbol{G}$	式（4-1）

第二步.网关上电时，选取一个随机的大素数r,通过式（4-2）计算EC上的点R作为网关的公钥，r作为网关的私钥。

$\mathbf{R} = r\boldsymbol{G}$	式（4-2）

第三步.网关初始化公钥私钥完成后，通过TCP协议连接云服务器，经过TCP三次握手建立连接，之后网关向云服务器发起认证请求，将网关的公钥R和网关ID发送到云服务器.

第四步.云服务器接收到网关的认证请求之后，首先验证网关ID的合法性,之后用一个hash映射保存网关的ID和公钥，用做后面数字签名的验签。云服务器使用自己的私钥和网关的公钥通过公式（4-3）计算出AES加密密钥K。然后将自己的公钥P发送给网关。

$\boxed{K = p \cdot R = p \cdot r \cdot G}$	式（4-3）

第五步.网关接收到云服务器的公钥之后，使用自己的私钥和云服务器的公钥，通过公式（4-4）计算AES加密密钥K。

$\boxed{K = r \cdot P = r \cdot p \cdot G}$	式（4-4）

第六步.网关使用密钥K加密登录口令并发送给云服务器，验证密钥的正确性。

第七步.云服务器接收到网关登录口令之后，使用密钥K进行解密，然后与事先约定的登录口令进行对比，如果不一致就关闭连接并删除密钥K，如果一致就把与网关的会话加入连接管理，保存密钥K并向网关发送登陆成功的消息。

第八步.网关接收到登陆成功的消息之后，保存密钥K，后续使用密钥K进行AES加密解密通信。

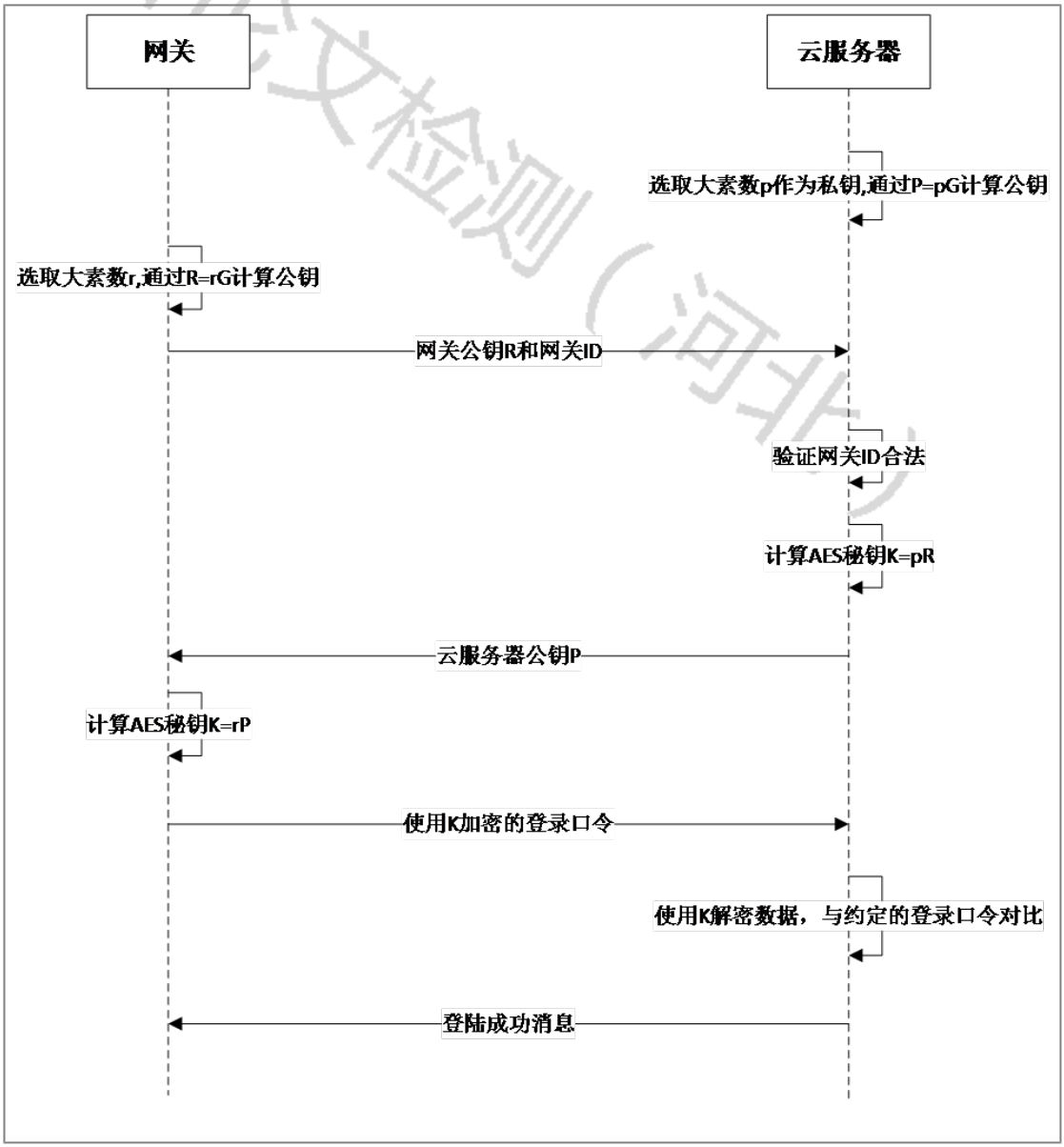


图4-1 网关与云服务器密钥协商

4.3.2 数字签名与数据加密

由文献[11]中几种数字签名算法的分析比较可以得出结论，与RSA和DSA相比，无论是在安全性、运算速度和存储空间上，ECC算法都有着明显的优势。现代计算机运算速度快，存储空间大，可以很好地应用RSA和DSA算法，但是在本文研究的农业物联网嵌入式设备中，网关的计算资源有限，ECC算法更加轻量级的特点使之成为更理想的选择。

签名和加密的流程如下：

1.发送方选择一个随机数k，k的范围是{1,...n-1}，n为基点G的阶。

2.发送方计算P = kG，G是基点。

3.发送方通过式（4-5）计算数字r,xp 是P的x轴坐标，如果r=0，选择另一个k重新计算。

$r = x_p \bmod n$	式（4-5）

4.计算明文摘要z = SHA1(m)，m是消息的明文。

5.通过式（4-6）计算s，其中dA是发送方的私钥，k-1是k mod n的乘法逆元。如果s=0，选择另一个k重新计算。

$s = k^{-1}(z + rd_A) \bmod n$	式（4-6）

6.使用AES128加密算法和协商出的密钥，对明文（m）进行加密，得出密文（w）。

7.将消息的签名（r，s）与密文（w）发送给接收方

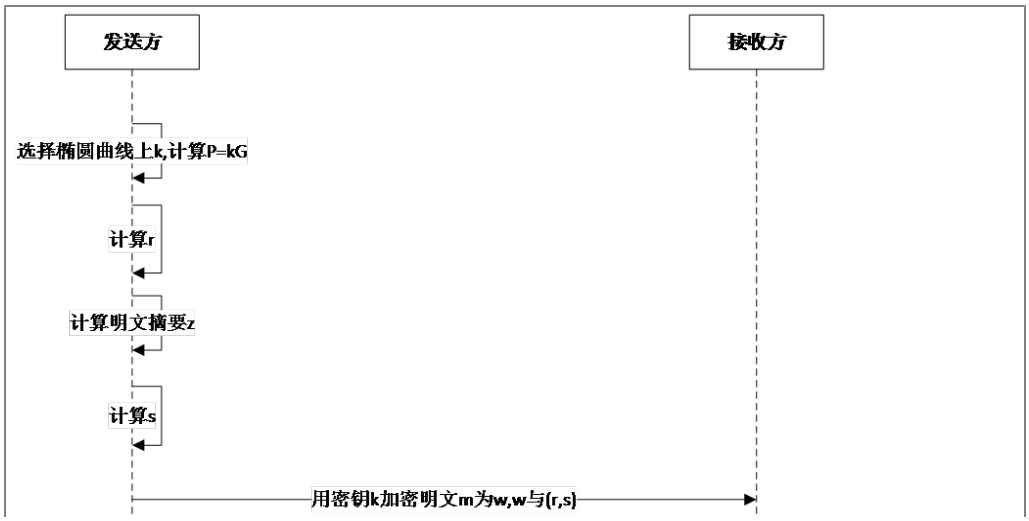
解密和验签的流程如下

1.接收方在收到密文（w）和签名值（r，s）之后，使用AES128解密算法和之前协商出的密钥对密文进行解密，得到明文（m）。

2.计算：sG+H(m)P = (x1,y1)，r1 = x1 mod p。

3.验证：r1 = r mod p。

4.如果等式成立，说明验签成功，数据没有被篡改过，否则签名无效，丢弃数据



维普论文检测（河北）

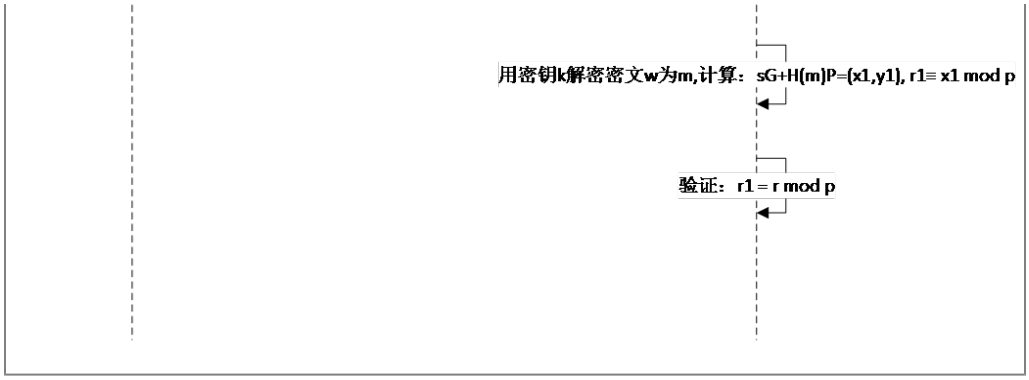


图 4-2 数字签名与验签

4.3.3 密钥更新

在安全的通信系统中，如果长期使用同一密钥，泄漏的可能性就会提高，通信系统的安全性就会下降。为了应对这个问题，需要系统中增加密钥更新机制。本系统中，可以通过web端登录用户管理中心，在网关列表选择网关进行密钥更新。密钥更新的流程如下：

1.用户登录云服务器

2.用户查询在线网关列表中的网关

3.用户选择指定的网关，向云服务器发送更新网关密钥的请求

4.云服务器验证用户和网关的合法性之后，向网关发送密钥更新命令

5.网关接收到密钥更新命令之后，重新执行密钥协商和网关登录

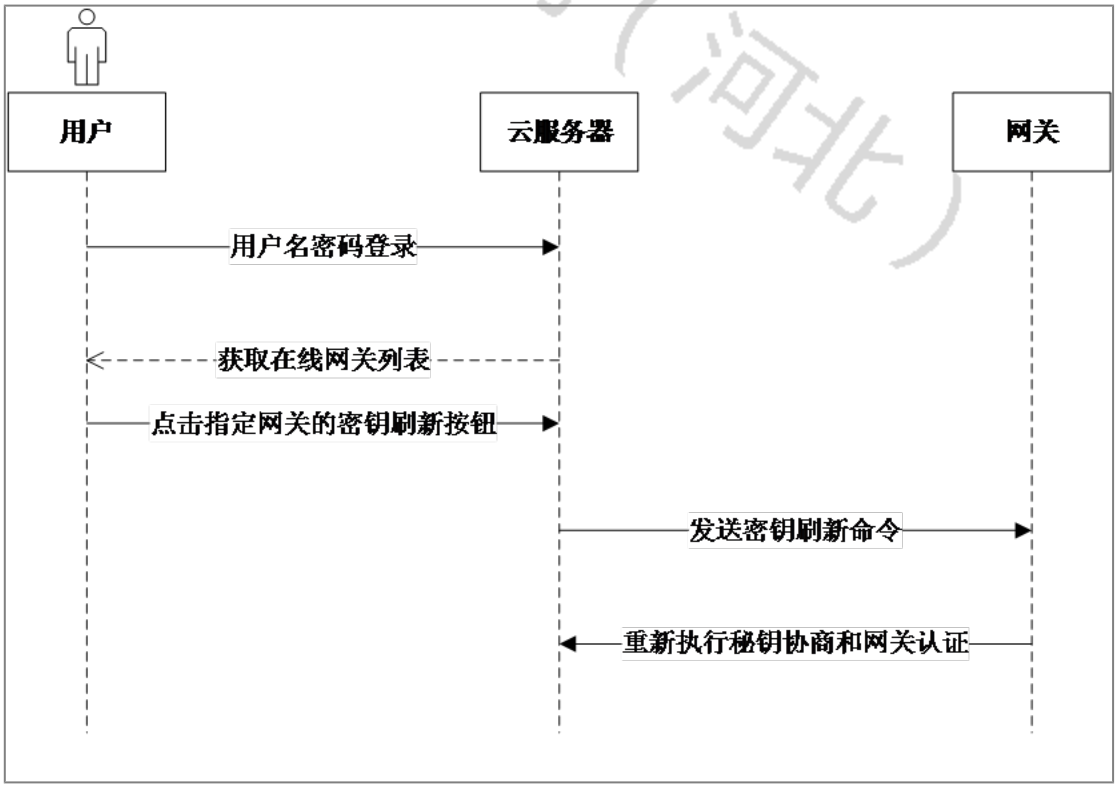


图 4-3 用户刷新网关密钥

4.4 算法安全性分析

1. 防窃听分析

云服务器与网关之间建立通信信道时，监听者可以获取公钥Kc-g，但无法得到双方的私钥k，即使获取密文C，也无法获取真实信息，

因此本方案具有防窃听能力。

3.防假冒分析。

系统中通信信息使用存有椭圆曲线 Ep(a,b) 、选定的基点G 和私钥t ，

其中私钥是保存在本地，不会在网络中传输，并且使用STM32芯片的读保护模式，使用芯片的全球唯一id作为网关ID，启动时与代码中的ID进行校验，一旦不同则进行flash数据擦除。窃听者无法伪造加密的信息，因此在本方案具有防假冒能力。

4.防重传分析。

重传攻击是窃听者在非法获取到网络中的通讯信息后，通过重发先前信息以非法获取信任的攻击手段。本系统消息中添加了时间戳、有效期和顺序号等信息，使得本方案具有防重传的能力。

5.中间人攻击

中间人攻击是攻击者分别于通信双方建立连接，然后窃听或修改通信双方信息的攻击手段。在本系统中，由于存在双向认证和数字签名的机制，攻击者无法伪造出合法的认证信息，所以本放安可以防止中间人攻击。

第5章系统测试

本章首先介绍了测试环境，从实现的功能和系统性能两个方面，设计了测试用例，对系统进行了全面的测试。

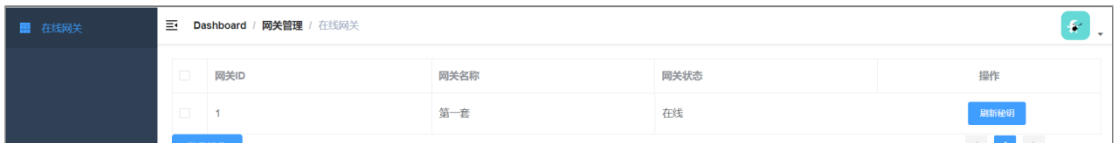
5.1测试环境

本文在设计和实现物联网网关通信实验仿真过程中主要是通过在PC电脑上使用Java程序模拟双向认证及安全通信的过程。整个系统分为云服务器端、Web客户端、网关客户端三个部分，其中，云服务器端和网关客户端使用Java语言为开发语言，版本为Java8，使用开源的Netty作为网络通信的框架，实现Socket服务端的网络数据监听，使用bouncycastle作为Java加密解密库的扩展，实现了本文描述的ECC算法。Web客户端使用JavaScript开发语言和开源的Vue.js前端框架。整个系统运行在PC计算机上，该计算机使用64位windows操作系统（CPU i5-7200U，2.50GHz、内存8GB、磁盘空间128GB）。

5.2 系统功能测试

整个设计方案所要实现的最终功能是能够实现物联网管关和云服务器之间密钥协商和数据加密传输，并通过Web页面进行在线网关和密钥的管理。本系统的综合测试中，首先要对密钥协商和双向认证功能进行测试，协商出密钥之后，再对数据加密解密和数字签名进行测试，最后通过Web端登录后台管理中心，进行密钥更新测试。Web端页面截图如下：

图 5-1 系统登录页



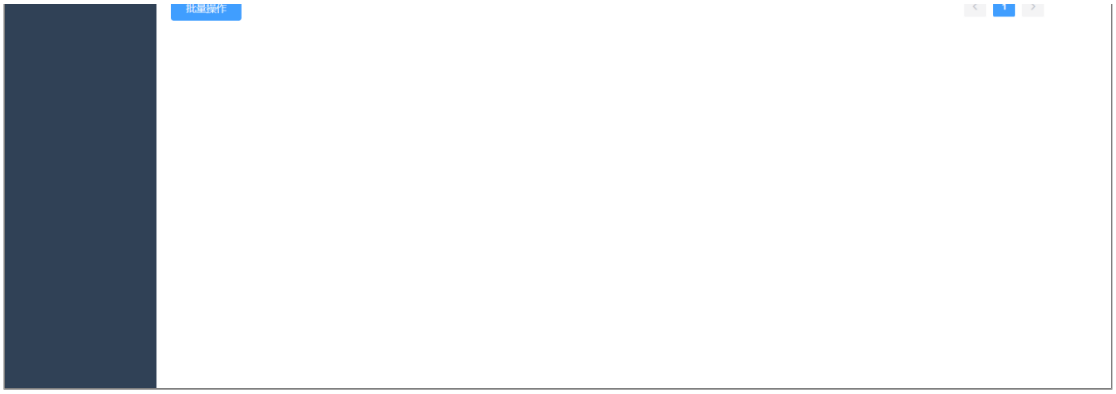


图 5-2 系统首页

5.2.1 密钥协商测试

在PC机上部署服务端，并启动网关客户端，在日志中打印网关与服务端进行密钥协商的过程。经过多次测试，在第一次进行密钥生成和密钥协商时，时间耗时在1400ms左右，后面刷新密钥时，再次生成密钥和进行密钥协商，耗时在40ms左右，速度较快，对主要逻辑影响较小。运行截图如下：

```
: No active profile set, falling back to default profiles: default
: Bean 'org.springframework.hateoas.config.HateoasConfiguration' of type [org.springframework.hateoas.config.HateoasConfigu
: Tomcat initialized with port(s): 8081 (http)
: Starting service [Tomcat]
: Starting Servlet engine: [Apache Tomcat/9.0.17]
: Initializing Spring embedded WebApplicationContext
: Root WebApplicationContext: initialization completed in 3985 ms
: 启动 Netty 成功
: 服务端初始化公钥serPubKey【MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAET73HUlb+anKRCY/uBSgbwCMhW97/s8YbcqeDUpKO

: 服务端初始化私钥serPriKey【MIGTAgEAMBMGByqGSM49AgEGCCqGSM49AwEHBHkwdwIBAQQgILfcmnPIihV6Hu4lwvSk/v5Uaz5B
```

```
: 启动 Netty客户端 成功
: Initializing ExecutorService 'applicationTaskExecutor'
: 【客户端初始化公钥cliPubKey】MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEYerboK2RTxDdQmPO0iSuzGzT3XW6NeV6KIZYnmja

: 【客户端初始化私钥cliPriKey】MIGTAgEAMBMGByqGSM49AgEGCCqGSM49AwEHBHkwdwIBAQQgfJv0PpldCKEImF4BDDcj0zQzos+D

: 1.客户端发起秘钥协商请求
: 连接Netty服务端成功
: Tomcat started on port(s): 8082 (http) with context path ''
: Started ClientApp in 7.875 seconds (JVM running for 10.915)
: 4.客户端开始秘钥协商，收到服务端公钥为:MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAET73HUlb+anKRCY/uBSgbwCMhW97/s8YbcqeDUpKO

: 5. 客户端秘钥协商完成，开始验证秘钥正确性,加密秘钥为:sAe3R+/e59Ho0rCkt+CMVEn77K7q0ChMoHHOh9FU4Fw=
: 8. 网关登陆成功，后续数据将使用AES进行加解密
```

```
: 客户端【/127.0.0.1:58007】连接，开始认证
: 2.服务端接收到秘钥协商请求,客户端公钥为：MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEYerboK2RTxDdQmPO0iSuzGzT3XW6NeV6KIZYnmja

: 3.服务端协商出加密秘钥：sAe3R+/e59Ho0rCkt+CMVEn77K7q0ChMoHHOh9FU4Fw=
: 6.服务端验证秘钥正确性
: 7. 服务端解密出登录口令:login
: 网关【1】验证通过，加入连接列表
```

图 5-3 密钥协商过程

测试用例如下：

表 5-1 密钥协商测试用例

用例描述	操作步骤	预期结果	实际结果
网关与云服务器进行密钥协商	1在PC机部署并初始化云服务器 2启动客户端程序 3 重复运行10次	网关与云服务器协商出相同的密钥，云服务器管理网关连接列表。	符合预期

5.2.2 数字签名和加密解密测试

网关与云服务器协商出加密密钥，后续发送和就收数据都会使用AES128算法进行加密解密。同时，在通信过程中，为了保证消息不会被中间人篡改，通信双方都会保存对方的公钥，并且在发送数据时使用通过SHA算法对明文生成摘要，使用自己的私钥对摘要进行SHA1withECDSA数字签名；接收方接收到数据时，先使用对方的公钥验证数字签名，如果验签失败则直接丢弃数据，如果验签成功，说明数据没有被篡改，进行后续处理。运行截图如下：

图 5-4 数字签名和数据加密解密

测试用例如下：

表 5-2 数字签名和数据加密解密用例

用例描述	操作步骤	预期结果	实际结果
网关与云服务器通信时对消息进行加密解密和数字签名验签	1在PC机部署并初始化云服务器 2启动客户端程序 3 重复运行10次	网关向云服务器发送经过AES加密和数字签名过的数据，云服务器AES解密数据和验签。云服务器向网关发送数据。	符合预期

5.2.3 心跳管理测试

网关与云服务器双向认证通过后，会加入心跳连接的机制，通过向对方发送心跳和心跳应答，确认对方是否在线。在云服务器端维护一个计数器，当网关三次心跳都没有应答时，判断网关下线，关闭服务器与网关的TCP连接。在网关的逻辑是，云服务器宕机连接被断开时，会启动一个定时任务，每30秒进行重连。运行截图如下：

```
.JsonEncoderAES  : 加密前的明文为:{"content":"ping","id":1,"type":4}
.JsonEncoderAES  : 加密耗时：0，加密后的明文为:7D2DA92C623B9DE902B75106581DF5E00FCE69083C543A63D0C8F2C71DE615CED480E78019CE8E14D9C9FC2F23B6DDD3
.JsonDecoderAES  : 接收到密文为:7D2DA92C623B9DE902B75106581DF5E010197300880608C7B0A95B5287F38EE45754350A0B6D0B3F4AF04929FEED2C31D480E78019CE8E
.JsonDecoderAES  : 解密耗时：0，解密出的明文为:{"content":"ping","gatewayId":"1","id":0,"type":4}
.JsonEncoderAES  : 加密前的明文为:{"content":"pong","id":1,"type":5}
.JsonEncoderAES  : 加密耗时：0，加密后的明文为:29003EDB92B775EEA2E0A3C32729202F0FCE69083C543A63D0C8F2C71DE615CE714A0296D2DFDAC0E1688A565CD762B8
.JsonEncoderAES  : 加密前的明文为:{"content":"ping","id":1,"type":4}
.JsonEncoderAES  : 加密耗时：0，加密后的明文为:7D2DA92C623B9DE902B75106581DF5E00FCE69083C543A63D0C8F2C71DE615CED480E78019CE8E14D9C9FC2F23B6DDD3
.JsonEncoderAES  : 加密前的明文为:{"content":"ping","id":1,"type":4}
.JsonEncoderAES  : 加密耗时：0，加密后的明文为:7D2DA92C623B9DE902B75106581DF5E00FCE69083C543A63D0C8F2C71DE615CED480E78019CE8E14D9C9FC2F23B6DDD3
.JsonEncoderAES  : 加密前的明文为:{"content":"ping","id":1,"type":4}
.JsonEncoderAES  : 加密耗时：0，加密后的明文为:7D2DA92C623B9DE902B75106581DF5E00FCE69083C543A63D0C8F2C71DE615CED480E78019CE8E14D9C9FC2F23B6DDD3
rHeartHandler    : 网关3次心跳没有应答，判断下线，关闭通道
rHeartHandler    : 客户端【/127.0.0.1:55799】断开连接
```

图 5-5 心跳管理

测试用例如下：

表 5-3 心跳测试用例

用例描述	操作步骤	预期结果	实际结果
网关与云服务器之间维持心跳连接	1在PC机部署并初始化云服务器 2启动客户端程序 3重复运行10次	云服务器和网关检测socket读写空闲，空闲时向对方发送心跳，接收到心跳时发送应答	符合预期
网关心跳无应答，云服务器将其标为下线	1在PC机部署并初始化云服务器 2启动客户端程序 3阻塞客户端程序 4重复运行10次	云服务器发送三次心跳，都没有收到应答之后，将网关标记下线，关闭通信信道	符合预期
云服务器宕机，网关执行重连	1在PC机部署并初始化云服务器 2启动客户端程序 3手动关闭服务端程序 4重复运行10次	网关检测到TCP连接断开，判断云服务器宕机，执行定时重连	符合预期

5.2.4 密钥更新测试

用户在登录物联网网关轻量级认证加密方案用户管理中心后，点击在线网关模块，列表显示当前在线的网关，用户可点击网关后面的刷新密钥按钮，刷新指定网关的密钥。程序截图如下：

图 5-6 在线网关页面

图 5-7 密钥更新运行截图

测试用例如下：

表 5-4 密钥更新用例

用例描述	操作步骤	预期结果	实际结果
用户刷新指定网关的密钥	1用户登录物联网网关轻量级认证加密方案用户管理中心 2点击在线网关模块 3选择网关，点击密钥刷新按钮 4 重复运行10次	提示密钥刷新成功，查看后台日志，密钥已经被更新	符合预期

5.3 系统性能测试

5.3.1 并发压力测试

现实世界中物联网网关的数量是成千上万的，云服务器需要能够支持大量的物联网网关并发地进行连接和数据发送，这对于云服务器的设计十分具有挑战性。本文设计的云服务器使用NIO的IO模型，单一线程就可以支持多个客户端连接，同时使用多线程技术，异步地执行逻辑。

测试用例如下：

表 5-5 并发测试用例

用例描述	操作步骤	预期结果	实际结果
云服务器并发压力测试	1在PC机部署并初始化云服务器 2 同时启动多个网关客户端进行身份认证和登录	云服务器能够快速响应，并且不会宕机	符合预期

测试结果如下：

表 5-6 并发压力测试结果

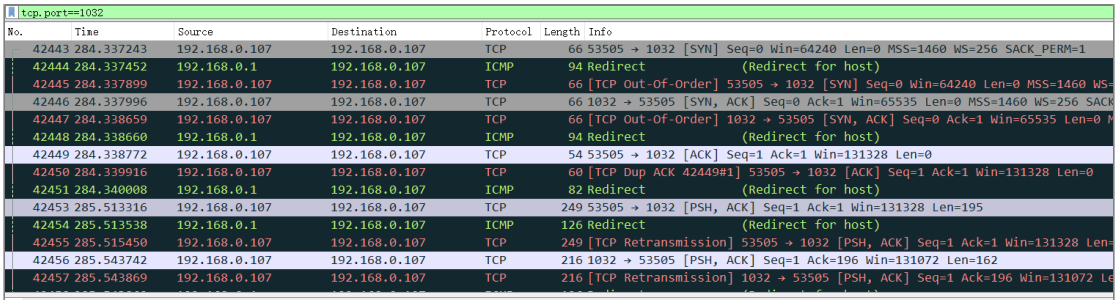
客户端并发连接数	服务端认证总时间	实际测试情况
10	154ms	服务器快速响应，没有卡顿现象
50	240ms	服务器快速响应，没有卡顿现象
100	512ms	服务器快速响应，没有卡顿现象
200	941ms	服务器响应较慢，没有卡顿
500	3102ms	服务器响应较慢，没有卡顿
1000	21043ms	服务器响应较慢，卡顿
3000	--	服务器出现拒绝连接的情况

测试结果：当前是在windows系统中进行测试，测试结果受CPU，内存，端口数量等限制。单机部署的情况下，合适的并发连接数在300左右，如果对性能要求较高，可以通过分布式部署、负载均衡等技术解决。

5.3.2 安全性测试

在云服务器与网关通信的过程中，进行网络抓包，截获通信过程中的数据包并进行分析，测试能否通过数据包中的信息破解出网关的密钥。

运行截图如下：



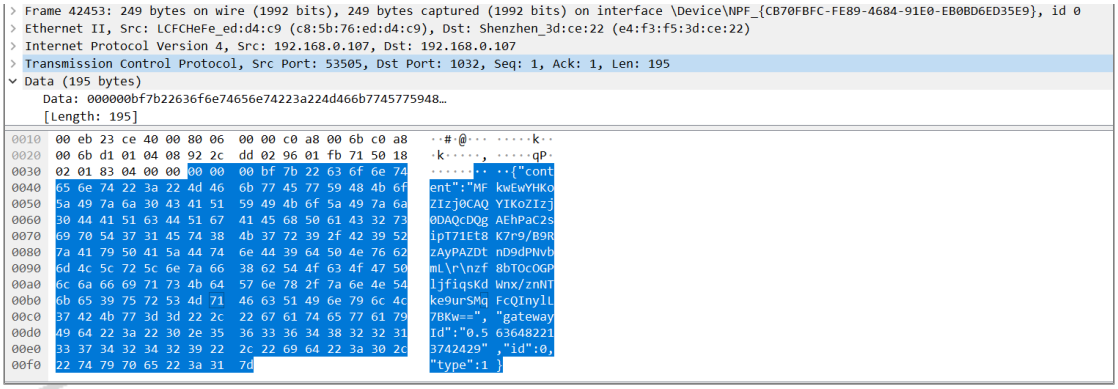


图 5-8 安全性测试抓包截图

测试用例如下：

表 5-7 安全性测试用例

用例描述	操作步骤	预期结果	实际结果
使用wireshark软件抓包，分析抓取的数据包	1部署云服务器端和网关客户端 2在通信过程中使用wireshark抓包 3分析抓取的数据包，看能否通过数据包破解密钥	从网络数据包中无法破解出密钥	符合预期

第六章结论

本文研究了在农业物联网的环境下，基于椭圆曲线ECC和AES128的物联网网关轻量级认证与加密方案，该方案的原理主要是利用椭圆曲线的密钥协商算法（ECDH）和数字签名算法（ECDSA），使用AES128算法进行对称加密通信。系统实现的技术支持为Java语言、Netty网络通信框架、Spring Boot框架开发实现的。系统中实现了物联网网关与云服务器之间进行密钥协商、非阻塞、IO多路复用的通信模型、数字签名、心跳管理、密钥刷新等功能。可以作为一个高性能、轻量级的物联网认证与加密方案，该算法运行时占用资源较少，适合在嵌入式设备上运行，对具体的业务逻辑影响较小。

论文工作总结：

(1)需求分析与系统设计，阅读相关文献，调查物联网认证和加密系统的基本架构，分析了将椭圆曲线算法落实到实际项目的可行性，最终通过编码实现了利用ECC和AES的轻量级认证和加密方案。

(2)通过设计基于椭圆曲线ECC和AES128的物联网网关轻量级认证与加密方案，学到了一个软件系统从需求分析、数据结构设计、算法设计到最后编码实现等整体的流程，在这其中也遇到了很多的问题，例如如何解决TCP粘包拆包、Netty框架中如何动态加入加密解密的编解码器等，这些都是在平时课堂上学不到的，编码能力和解决问题的能力有了明显的提高。

(3)目前基于椭圆曲线ECC和AES128的物联网网关轻量级认证与加密方案的代码已按时开发完成，但是系统仍然有优化的空间，例如可以通过添加定时任务的方式，每过一段时间就去刷新密钥，而不需要人在页面上点击；本文中为了统一技术框架和更好的可视化实验，网关客户端的逻辑是用Java语言实现的，在实际的应用中，网关客户端应该使用真正的嵌入式设备；另外可以接入移动客户端，加入节点远程控制、数据监测等功能。

谢辞

行文至此，已经到了论文的最后一步，四年的大学生涯即将结束。学生时代的生活是弥足珍贵的，因为时光是一张有去无返的单程

维普论文检测（河北）

票，关于北方学院有我带不走的记忆，更有带不走的收获。大学生活中最有价值的一件事就是在大二的时候加入信息技术研究所，在这里遇到了志同道合的老师和同学，学到了很多实际项目中的技术。

首先，感谢我的指导老师郭本振老师和科研室的王志辉老师。两位老师认真的工作态度和科学的引导给了我很大的帮助，没有两位老师的悉心指导，就没有我的这个毕业设计。两位老师在项目设计期间，给出了许多中肯的建议，在论文修改期间，针对论文中的不足之处悉心指导，在此我深表敬意。

其次，还要对大学期间每位任课老师和同学表达谢意，是你们陪我度过了人生中宝贵的四年，你们的帮助和陪伴使我不断成长和进步，让我度过了充实且快乐的大学时光。

最后感谢各位评审老师，能来参加我的毕业答辩，给我提出珍贵的建议，在此表示深深的感谢。

参考文献

[1] 汪洋. 物联网轻量级认证和加密技术研究[D].南京邮电大学,2017.

[2] Nicanfar H, Leung V C M. Multilayer Consensus ECC-Based Password Authenticated Key-Exchange (MCEPAK) Protocol for Smart Grid System[J]. IEEE Transactions on Smart Grid, 2013, 4(1):253-264

[3] Li D, Aung Z, Williams J R, et al. Efficient and fault-diagnosable authentication architecture for AMI in smart grid[J]. Security & Communication Networks, 2015, 8(4):598-616.

[4] Q. Jiang, Y. Qian, J. Ma, X. Ma, Q. Cheng, and F. Wei, “User centric three-factor authentication protocol for cloud-assisted wearable devices,” Int J Commun Syst, vol. 32, no. 6, p. e3900, Apr. 2019.

[5] ZHIHUI WANG, JIANLI ZHAO, BENZHEN GUO, JINGJINGYANG, XIAO ZHANG. Mutual Authentication Protocol for IoT-based Environment Monitoring System[j]Journal of Environmental Protection and Ecology 2019,6(2)

[6] 袁琦. 我国物联网安全发展现状和建议[J]. 现代电信科技, 2014(10):36-39.

[7] 史冰清. 高安全性的物联网网关设计与实现[D].电子科技大学,2018.

[8] 王斌. 工业物联网信息安全防护技术研究[D].电子科技大学,2018.

[9] 黎俊男. 基于AES与ECC的游戏数据混合加密研究与实现[D].华南理工大学,2018.

[10]龙辉. 基于ECC-AES混合加密的智能配电网安全通信方案设计[D].湘潭大学,2016.

[11]李春平. 基于低功耗处理器的数字签名研究与实现[D].北京邮电大学,2015.

[12]于庆文. 谈密码技术的发展趋势[J]. 网络与信息,2012,26(09):76-77.

[13]李栋. 无线传感器网络中能量优化与安全方案研究[D].北京邮电大学,2013.

[14]严佳韵. 基于椭圆曲线的快速数字签名算法[D].西南交通大学,2011.

[15]刘佳. 基于ECC与嵌入式的智能家居安全协议研究与设计[D].电子科技大学,2017.

[16]田兴宇. 基于STM32的嵌入式加密链路机的设计与实现[D].黑龙江大学,2018.

[17]刘克一. 物联网概念的基本定位[J]. 数字技术与应用,2015(04):221.

[18]刘欣东,徐水帅,陈建华. 基于椭圆曲线密码的智能电网通信认证协议[J]. 计算机应用,2019,39(03):779-783.

附录

```
@Slf4j
public class ServerAuthHandler extends ChannelInboundHandlerAdapter {
@Override
public void channelActive(ChannelHandlerContext ctx) {
log.info("客户端【{}】连接，开始认证",ctx.channel().remoteAddress());
}
@Override
```

```
public void channelRead(ChannelHandlerContext ctx, Object message) {
SocketMsg msg = (SocketMsg) message;
try {
switch (msg.getType()){
case MsgType.AUTH_VALUE:
String clientPubKey = (String) msg.getContent();
String gwId = msg.getGatewayId();
if (StringUtils.isEmpty(clientPubKey) || StringUtils.isEmpty(gwId)){
log.info("网关公钥或网关ID为空，拒绝处理");
closeChannel(ctx);
return;
}
long start = System.currentTimeMillis();
log.info("2.服务端接收到密钥协商请求,客户端公钥为：{}",clientPubKey);
EcKeys ecKeys = SpringBeanFactory.getBean("EcKeys",EcKeys.class);
SocketMsg<String> publicKeyStr = new SocketMsg<String>()
.setId(1).setType(MsgType.AUTH_BACK_VALUE).setContent(ecKeys.getPubKey());
String key = EncryptOrDecryptUtil.ecdhKey(ecKeys.getPriKey(),clientPubKey);
log.info("3. 服务端密钥协商完成，耗时：{}ms，开始验证密钥正确性,加密密钥为:{}",System.currentTimeMillis()-start,
key);
//将改密钥加入密钥库
Gateway gw = new Gateway()
.setGwId(msg.getGatewayId())
.setGwName("第一套")
.setChannel(ctx.channel())
.setKey(key)
.setPubKey(clientPubKey);
NettySocketHolder.put(msg.getGatewayId(), gw);
ctx.writeAndFlush(publicKeyStr);
break;
case MsgType.AUTH_CHECK_VALUE:
log.info("6.服务端验证密钥正确性");
Gateway gw1 = NettySocketHolder.get(msg.getGatewayId());
//确认密钥的正确性
String login = EncryptOrDecryptUtil.doAES((String) msg.getContent(),gw1.getKey(), Cipher.DECRYPT_MODE);
log.info("7.服务端解密出登录口令:{}",login);
if (("login").equals(login)){
log.info("网关【{}】验证通过，加入连接列表",msg.getGatewayId());
gw1.setStatus("在线");
```

维普论文检测（河北）

```
NettySocketHolder.put(msg.getGatewayId(),gw1);
//向客户端发送认证成功的消息，这里一定要先发消息再替换handler
ctx.writeAndFlush(new SocketMsg<String>().setType(MsgType.LOGIN_SUCCESS_VALUE));
//用AES加解密替换掉默认的编解码器
ctx.pipeline().replace(ctx.pipeline().get("decoder"),"decoder",new JsonDecoderAES());
ctx.pipeline().replace(ctx.pipeline().get("encoder"),"encoder",new JsonEncoderAES());
ctx.pipeline().addFirst("idle",new IdleStateHandler(5,0, 0, TimeUnit.SECONDS));
}else {
log.info("非法客户端，关闭连接");
closeChannel(ctx);
}
break;
default:break;
}
}catch (Exception e){
e.printStackTrace();
log.error("客户端认证出错{}",e.getMessage());
closeChannel(ctx);
}
ctx.fireChannelRead(message);
ReferenceCountUtil.release(msg);
}
private void closeChannel(ChannelHandlerContext ctx){
NettySocketHolder.remove(ctx.channel());
ctx.channel().close();
}
}
```

- **说明：**

相似片段中"综合"包括：

《中文主要报纸全文数据库》 《中国专利特色数据库》 《中国主要会议论文特色数据库》 《港澳台文献资源》 《图书资源》 《维普优先出版论文全文数据库》 《年鉴资源》 《古籍文献资源》 《IPUB原创作品》

- **声明：**

报告编号系送检论文检测报告在本系统中的唯一编号。

本报告为维普论文检测系统算法自动生成，仅对您所选择比对资源范围内检验结果负责，仅供参考。

维普论文检测（河北）

客服热线：400-607-5550 | 客服QQ：4006075550 | 客服邮箱：vpcs@cqvip.com

唯一官方网站：http://vpcs.cqvip.com

关注微信公众号