计算机科学
COMPUTER SCIENCE

基于国密算法的MQTT安全机制研究与实现

刘泽超, 梁涛, 孙若尘, 郝志强, 李俊

引用本文

刘泽超, 梁涛, 孙若尘, 郝志强, 李俊. 基于国密算法的MQTT安全机制研究与实现[J]. 计算机科学, 2024, 51(2): 333-342.

LIU Zechao, LIANG Tao, SUN Ruochen, HAO Zhiqiang, LI Jun. Research and Implementation of MQTT Security Mechanism Based on Domestic CryptographicAlgorithms [J]. Computer Science, 2024, 51(2): 333-342.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.221100157

基于国密算法的 MQTT 安全机制研究与实现

刘泽超[1,2]　梁　涛[1]　孙若尘[1]　郝志强[3,4]　李　俊[3,4]
1 哈尔滨工程大学计算机科学与技术学院　哈尔滨 150001
2 电子政务建模仿真国家工程实验室　北京 100037
3 国家工业信息安全发展研究中心　北京 100040
4 三亚学院信息与智能工程学院　海南 三亚 572022
(liuzechao@hrbeu.edu.cn)

摘　要　针对现有 MQTT 协议缺乏有效身份认证以及数据以明文形式传输的问题,提出了一种基于国密算法 SM2,SM3,SM4 的 MQTT 安全保护方案。通过 SM2 算法实现客户端与 MQTT Broker 之间的双向身份认证;通过 SM4 算法加密 MQTT 协议中用户名、密码、主题的消息内容等数据;通过 SM3 算法保证 MQTT 协议传输数据的完整性。将自主可控的国产密码技术应用到 MQTT 协议中,可有效提升该协议的安全防护能力。安全性分析和实验结果表明,所提方案在解决了 MQTT 协议安全问题的同时,也可以满足实际的应用需求。

关键词:国密算法;MQTT 协议;身份认证;数据加密
中图分类号　TP309.2

Research and Implementation of MQTT Security Mechanism Based on Domestic Cryptographic Algorithms

LIU Zechao[1,2],LIANG Tao[1],SUN Ruochen[1],HAO Zhiqiang[3,4] and LI Jun[3,4]
1 College of Computer Science and Technology,Harbin Engineering University,Harbin 150001,China
2 Modeling and Emulation in E-Government National Engineering Laboratory,Beijing 100037,China
3 China National Industrial Information Security Development Research Center,Beijing,100040,China
4 School of Information & Intelligence Engineering,University of Sanya,Sanya,Hainan 572022,China

Abstract　Aiming at the problem that existing MQTT protocol lacks effective identity authentication and data plaintext transmission,an MQTT security protection scheme is designed based on domestic cryptography algorithms SM2,SM3 and SM4. Two-way identity authentication between the client and MQTT Broker is realized by SM2 algorithm. SM4 algorithm is used to encrypt the username,password,and message contents of subjects in MQTT protocol. SM3 algorithm is used to ensure the integrity of data transmitted by MQTT protocol. Applying self-controllable domestic cryptography technology to MQTT protocol can effectively improve the security protection capability of the protocol. The security analysis and experimental results show that the proposed scheme can not only solve the security problem of MQTT protocol,but also meet the practical application requirements.

Keywords　Domestic cryptographic algorithms,MQTT protocol,Identity authentication,Data encryption

1　引言

物联网(IoT,Internet of Things)技术经过数十年的飞速发展,已经被越来越多地应用于生活中的各个方面[1-2]。据统计,预计到 2030 年,连入物联网中的设备将达到 500 亿个[3]。随着物联网设备的迅速增加,物联网环境也变得愈来愈复杂。因此,适用于各种物联网环境的通信协议也相继被提出。消息队列遥测传输协议(Message Queuing Telemetry Transport,MQTT)是一种基于代理的发布/订阅模式的消息传输协议,属于 TCP/IP 协议之上的应用层协议,因具有轻量、简单、开放和易于实现的特点,适用于低功耗和网络带宽有限的 IoT 场景[4],已成为物联网中受欢迎的通信协议之一。

到稿日期:2022-11-19　返修日期:2023-12-15
基金项目:国家重点研发计划(2021YFB3101602);工信部产业基础再造和制造业高质量发展专项项目(TC220A04X-1);基础科研项目(JCKY2020604C011)
This work was supported by the National Key R & D Program of China(2021YFB3101602),Special Project for Industrial Foundation Reconstruction and High Quality Development of Manufacturing Industry by the Ministry of Industry and Information Technology(TC220A04X-1) and Basic Research Program(JCKY2020604C011).
通信作者:李俊(lijun@cisc-cert.org.cn)

MQTT 协议现已被广泛应用于智能医疗、智能家居、智能车辆等诸多方面[5-6]。

传统的 MQTT 协议存在两方面的安全问题：1)缺乏有效身份认证。MQTT Broker 可以通过用户名、密码对连接的客户端身份进行认证，然而用户名与密码都是以明文传输，攻击者可以轻易冒充真实用户身份。2)主题的消息内容通过明文传输。MQTT 协议的主题消息内容均以明文形式进行传输，导致该协议数据的机密性无法得到保障。为了提升 MQTT 协议的安全性，结构化信息标准促进组织(Organization for the Advancement of Structured Information Standards，OASIS)提出可以使用 SSL/TLS 来解决 MQTT 协议的安全问题。这种解决方案虽然提供了安全保护机制，但其中涉及的数字证书管理与验证等带来了大量的计算与通信开销。

为了提高国家在信息安全领域的自主可控能力，我国密码管理局认定和公布了国密算法标准及其应用规范。国密算法是我国在密码核心领域自主研发的一套数据加密处理系列算法，该算法涉及各种加密类型，其中包括公钥加密算法 SM2 和 SM9；对称加密算法 SM1，SM4 和 SM7；密码杂凑算法 SM3 等[7]。国家目前正大力推广自主可控的国密算法，国密算法的普及对提升我国网络信息安全与自主可控水平具有重要意义[8]。

近年来，国内外学者针对 MQTT 协议的安全性问题也进行了大量研究。2015 年，Singh 等[9]提出了一种基于 MQTT 协议的安全方案 SMQTT。该方案采用属性基加密技术实现对传输数据的机密保护，但是 SMQTT 的计算开销较大。2017 年，Bisne 等[10]提出了一种基于密钥策略属性基加密(Key Policy Attribute based Encryption，KP-ABE)和 AES 加密的 MQTT 协议安全方案，该方案同样存在计算开销较大的问题。同年，Bhawiyuga 等[11]提出了一种基于 token 的 MQTT 协议身份认证方案。该方案中存在一个外部的身份认证服务器，这就增加了额外的硬件开销，且 token 以明文形式传输，导致其机密性较弱。2018 年，Calabretta 等[12]提出了一种基于增强的口令认证密钥交换协议的 MQTT 安全通信方案，该方案实现了客户端与 MQTT Broker 的双向身份认证，但是需要 MQTT Broker 对主题的消息内容进行解密重加密，因此该方案并没有实现端到端的安全性。2019 年，Su 等[13]提出了一种 MQTT-TTS 方案，该方案可以根据不同的应用场景动态地选择不同的加密方式。2020 年，Chien 等[14]提出了一种兼容 MQTT-API 的物联网安全增强方案，该方案提供了身份认证和数据加密的功能。但是该方案增加了外部设备，且部分通信采用 SSL 加密，增加了额外的硬件开销且计算和通信开销均较大，难以适用于资源受限的 MQTT 协议。Potrino 等[15]提出通过椭圆曲线加密技术对 MQTT 协议的关键数据进行加密，防止数据被非法篡改、窃听。Sanjuan 等[16]提出了一种利用加密智能卡实现 MQTT 协议的身份认证与数据加密功能的方案。Patel 等[17]提出了一种先在客户端进行注册，再与 MQTT Broker 进行身份认证和密钥协商的 MQTT 安全框架。Amanlou 等[18]提出了一种基于 ECDHE 密钥协商和预共享密钥认证算法的 MQTT 身份认证方案。2021 年，Gu 等[19]提出了一种基于代理重加密的 MQTT 协议端到端安全方案，该方案实现了数据端到端的安全传输，但是缺乏客户端与 MQTT Broker 之间的身份认证的功能。同年，Spina 等[20]提出了一种轻量级端到端的 MQTT 协议安全方案，该方案实现了身份认证与数据加密的功能，但是在身份认证的过程中使用了数字证书，极大地增加了 MQTT 协议的计算开销。2022 年，Mendoza-Cardenas 等[21]将 CP-ABE 和 AES 算法应用于树莓派的 MQTT 协议中，保证了主题的消息内容的机密性。

虽然上述方案可以解决 MQTT 协议的部分安全问题，但是这些方案仍然存在计算开销大、机密性差等问题。因此，本文基于国密算法 SM2，SM3，SM4 提出一种轻量的 MQTT 协议安全保护方案。本文主要的研究工作如下：

1)提出了一种基于国密 SM2 的无证书双向身份认证方案，保证连接 MQTT Broker 中的客户端身份是可信任的。

2)将对称加密算法 SM4 和哈希算法 SM3 结合起来，对在 MQTT 协议中传输的数据进行加密保护与完整性验证，并实现端到端的机密传输。

3)安全性分析表明，所提方案除了能保证连接到 MQTT Broker 的客户端都是受信任的，以及防止恶意连接的客户端在 MQTT 协议中发布虚假消息外，还可对协议传输的重要数据进行机密保护，防止攻击者非法获取和篡改。

4)仿真实验结果表明，所提方案能够提高 MQTT 协议的安全性，同时其开销还可以满足 MQTT 协议的实际应用需求。

2 预备知识

2.1 国密算法 SM2

SM2 算法是国家密码管理局于 2010 年发布的椭圆曲线公钥密码算法。SM2 算法为使用者提供了数字签名、密钥协商和数据加密三方面的功能[22]。由于本文构建的方案主要采用 SM2 算法的签名验签功能对实体进行身份认证，因此这里重点描述 SM2 算法的签名验签过程。

SM2 签名验签过程如下。

1)设待签名的消息为 M，为了获取消息 M 的数字签名 (r,s)，作为签名者的用户 A 应通过自身私钥 (X,Y) 实现以下运算步骤：

A1：置 $\bar{M}=Z_A \parallel M$(Z_A 是关于用户 A 的可辨别标识、部分椭圆曲线系统参数和用户 A 公钥的杂凑值)。

A2：计算 $e=H_v(\bar{M})$ 并将 e 的数据类型按照规则转化成整数。

A3：用随机数发生器产生随机数 $K\in[1,n-1]$。

A4：计算椭圆曲线点 $(x_1,y_1)=[k]G$ 并将 x_1 的数据类型按照规则转换成整数。

A5：计算 $r=(e+x_1) \bmod n$，若 $r=0$ 或 $r+k=n$，则返回 A3。

A6：计算 $s=((1+d_A)^{-1}\cdot(k-r\cdot d_A)) \bmod n$，若 $s=0$，则返回 A3。

A7：将 r 和 s 按照对应规则转换成字节串，消息 M 的签名为 (r,s)。

2）为了检验收到的消息 M' 及其数字签名 (r',s')，作为验证者的用户 B 应通过签名者公钥 P_A 实现以下运算步骤。

B1：检验 $r' \in [1,n-1]$ 是否成立，若不成立则验证不通过。

B2：检验 $s' \in [1,n-1]$ 是否成立，若不成立则验证不通过。

B3：置 $\overline{M'}=Z_A \parallel M'$。

B4：计算 $e'=H_v(\overline{M'})$，按照规则将 e' 的数据类型转换为整数。

B5：按照规则将 r' 和 s' 的数据类型转换为整数，计算 $t=(r'+s') \bmod n$，若 $t=0$，则验证不通过。

B6：计算椭圆曲线点 $(x_1',y_1')=[s']G+[t]P_A$。

B7：将 x_1' 按照规则转换成整数，计算 $R=(e'+x_1') \bmod n$，若 $R=r'$，则验证通过，否则验证不通过。

2.2 国密算法 SM4

SM4 算法是我国在密码核心领域自主研发的对称加密算法，是国家密码管理局于 2012 年发布的。与国际主流对称加密算法 DES 和 AES 类似，SM4 算法是一种分组加密算法，其密钥长度与分组长度都是 128 bit[23]。SM4 算法由 32 次迭代运算和 1 次反序变换 R 组成。

设明文输入为 $(X_0,X_1,X_2,X_3) \in (Z_2^{32})^4$，密文输出为 $(Y_0,Y_1,Y_2,Y_3) \in (Z_2^{32})^4$，轮密钥为 $rk_i \in Z_2^{32}$，i=0，1，2，…，31，轮函数 $F(X_0,X_1,X_2,X_3,rk)=X_0 \oplus T(X_1 \oplus X_2 \oplus X_3 \oplus rk)$，其中 T 为合成置换函数。

1）SM4 加密算法的运算过程如下：

（1）32 次迭代运算

$$X_i+4=F(X_i,X_{i+1},X_{i+2},X_{i+3},rk_i),i=0,1,2,\cdots,31 \tag{1}$$

（2）反序变换

$$\begin{aligned} &R(X_{32},X_{33},X_{34},X_{35})=(X_{35},X_{34},X_{33},X_{32}) \\ &(Y_0,Y_1,Y_2,Y_3)=R(X_{32},X_{33},X_{34},X_{35}) \end{aligned} \tag{2}$$

2）SM4 解密算法与加密算法结构相同，不同的仅是轮密钥的使用顺序。解密时，使用轮密钥序列为 $(rk_{31},rk_{30},\cdots,rk_0)$。

3 MQTT 协议

MQTT 协议是基于代理的发布/订阅的轻量级协议，其系统架构如图 1 所示。

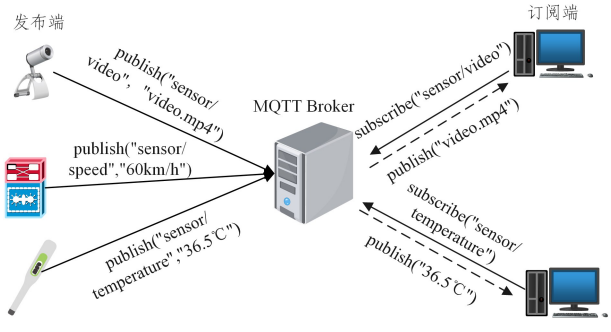

图 1 MQTT 系统架构

Fig. 1 MQTT system architecture

MQTT 协议由订阅端、发布端和 MQTT Broker 这 3 个实体组成，订阅端与发布端统称为客户端，MQTT Broker 为代理服务器。订阅端向 MQTT Broker 发送订阅主题，发布端向 MQTT Broker 发布主题的消息内容，MQTT Broker 则负责转发消息。

MQTT 协议为使用者提供了 3 种不同的服务质量来应对不同的应用场景[24]。其中 Qos0 代表消息至多发送一次，无论接收方是否接收到都不会再发第二次；QoS1 代表消息至少发送一次，如果发送方没有收到确认包，则会再次发送加上 DUP 标志的该消息，直到收到确认包为止；QoS2 代表消息始终只发一次，消息必须存储在发送方和接收方的本地环境中，直到它被妥善处理为止。本文所提方案适用于 MQTT 协议的所有服务质量。

和大多数通信协议不同的是，MQTT 协议有着自己的一套较为严格的数据交互流程[25]，如图 2 所示。

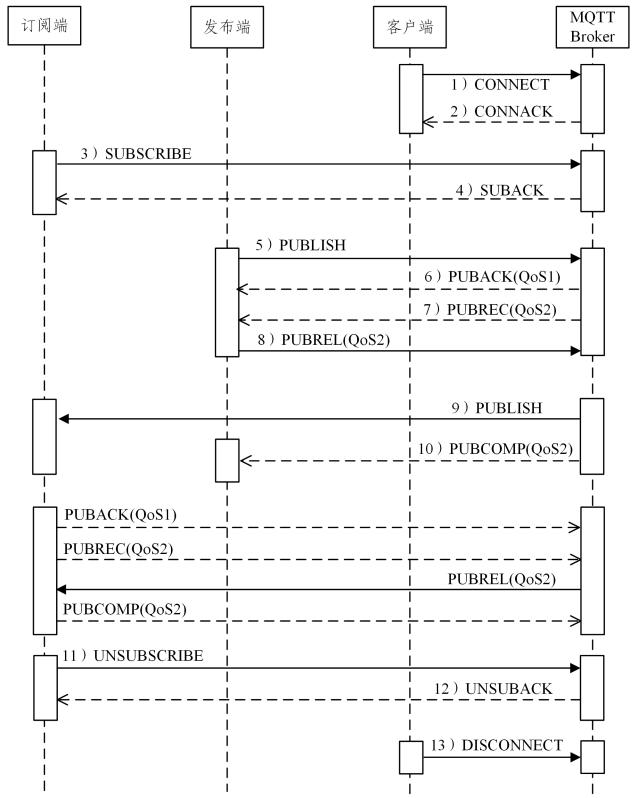

图 2 MQTT 时序图

Fig. 2 Sequence diagram of MQTT

1）客户端发送 CONNECT 数据包到 MQTT Broker 表示想要连接 MQTT Broker，其中 CONNECT 中包含用户名、密码等字段。

2）MQTT Broker 收到 CONNENT 数据包后通过用户名和密码验证身份，验证通过后返回 CONNACK 数据包来响应客户端。

3）如果订阅端收到 CONNACK 数据包，则其可以通过发送 SUBSCRIBE 数据包到 MQTT Broker 来订阅主题。

4）MQTT Broker 返回 SUBACK 数据包响应订阅端，SUBACK 中包含订阅是否成功等信息。

5）如果发布端收到 CONNACK 数据包，且服务质量为

QoS0，则 MQTT Broker 直接发送 PUBLISH 数据包到订阅端。PUBLISH 数据包中含有主题和主题的消息内容等信息。

6）如果服务质量为 QoS1，则发布端发送 PUBLISH 数据包后会在本地暂存这个 PUBLISH 数据包。MQTT Broker 收到 PUBLISH 数据包后，发送 PUBLISH 数据包到订阅端并返回 PUBACK 数据包到发布端，发布端收到 PUBACK 数据包后丢弃对应的 PUBLISH 数据包。

7）如果服务质量为 QoS2，则发布端发送 PUBLISH 数据包后会在本地暂存这个 PUBLISH 数据包。MQTT Broker 收到 PUBLSIH 数据包后在本地暂存这个 PUBLISH 数据包，然后返回 PUBREC 数据包。

8）发布端收到 PUBREC 数据包后暂存这个数据包并丢弃对应的 PUBLISH 数据包，然后发送 PUBREL 数据包到 MQTT Broker（QoS2）。

9）MQTT Broker 收到 PUBREL 数据包后将 PUBLISH 数据包发送到订阅端并丢弃本地存储的 PUBLISH 数据包，最后 MQTT Broker 返回 PUBCOMP 数据包到发布端（QoS2）。

10）发布端收到 PUBCOMP 数据包后丢弃之前存储的 PUBREC 数据包（QoS2）。

11）订阅端发送 UNSUBSCRIBE 数据包到 MQTT Broker，取消订阅主题。

12）MQTT Broker 返回 UNSUBACK 数据包，响应订阅端。

13）客户端发送 UNCONNECT 数据包到 MQTT Broker，断开与 MQTT Broker 的连接。

如图 3 所示，攻击者可以向 MQTT 协议发起攻击。

1）由于传统 MQTT 协议缺乏身份认证机制，故攻击者 B 可以冒充合法身份向 MQTT Broker 发送连接请求，一旦确认连接，攻击者将向 MQTT Broker 发送大量假消息，造成资源浪费。

2）由于传统 MQTT 协议的主题消息内容以明文的形式传输，故攻击者 A 可以向 MQTT 协议发起中间人攻击，截获主题消息内容，造成消息泄露。

3）由于传统 MQTT 协议缺乏数据完整性校验机制，故攻击者 A 在截获到主题消息内容后可以将其恶意篡改并重新发送至 MQTT 网络环境中，订阅端无法得知收到的消息是否被篡改。

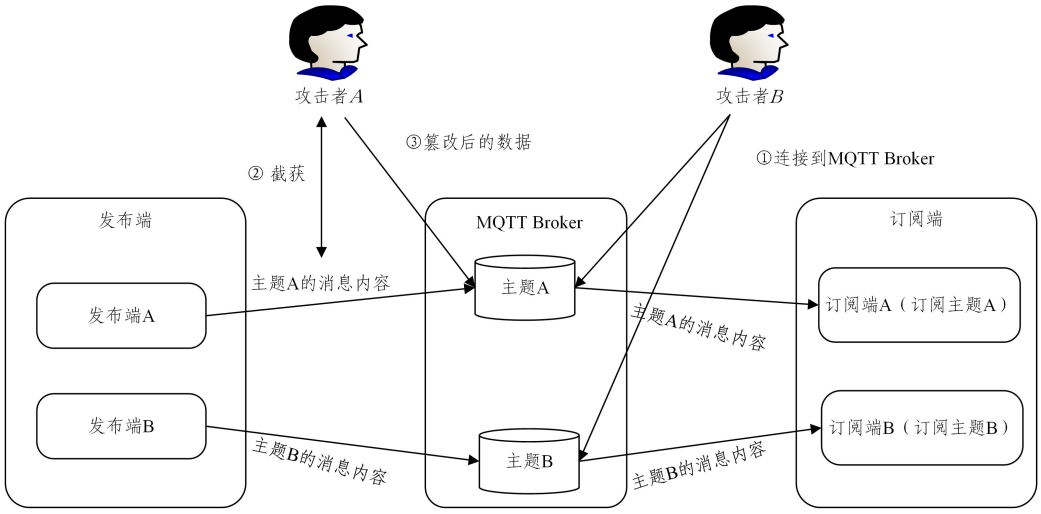

图 3 攻击示意图

Fig. 3 Attack diagram

为了解决 MQTT 协议的安全问题，本文提出的安全方案应满足以下几点要求：

1）客户端与 MQTT Broker 的双向身份认证。当客户端与 MQTT Broker 进行连接时，通过本文所提的安全方案，可以实现 MQTT Broker 对客户端的身份认证，同时客户端应当识别自己是否连接到目标 MQTT Broker。

2）端到端的安全性。在身份认证结束后要对 MQTT 协议中传输的主题的消息内容进行加密。为了提高数据的机密性，本文所提方案使得 MQTT Broker 仅对主题的消息内容做转发处理而无须解密，所以订阅端与发布端需要获得相同的关键密钥组件，组成相同的会话密钥，进而对主题的消息内容进行加密与解密。

3）轻量性。由于 MQTT 协议本身就是一个轻量、简单、易于实现的协议，因此本文所提方案在保证协议安全性的同时，所增加的开销也需要满足协议实际的应用需求。

4 MQTT 协议安全方案

4.1 方案总体描述

MQTT 协议安全方案共包含 4 种实体，分别是 MQTT Broker、发布端、订阅端与密钥生成中心（KGC）。其中 MQTT Broker 负责转发消息，发布端发布消息到 MQTT Broker，订阅端从 MQTT Broker 中接收消息，KGC 负责初始化系统参数。

本文所提方案可具体划分为 5 个阶段。1）初始化阶段：用于初始化系统公开参数。2）密钥生成阶段：生成 SM2 公私钥对，用于下一阶段的身份认证。3）身份认证阶段：用于客户端与 MQTT Broker 之间的双向身份认证。4）关键密钥组件获取阶段：用于获取发布端与订阅端之间的会话密钥组件，以便在下一阶段生成会话密钥。5）数据传输阶段：将 PUBLISH 数据包中的主题消息内容进行加密，并且通过哈希

函数保证数据的完整性，防止中间人攻击。为便于理解整体方案，图 4 给出了所提安全方案的时序过程。

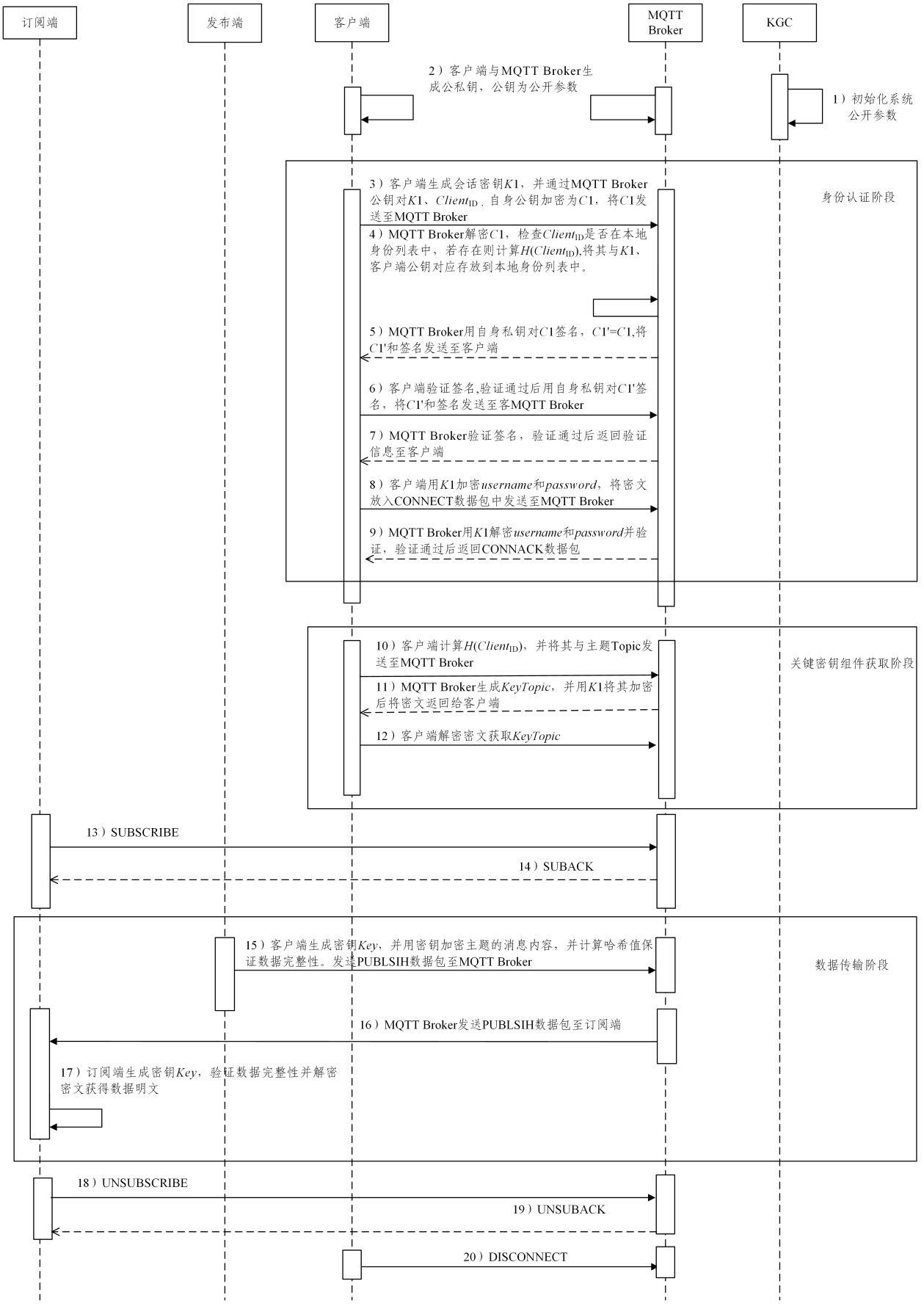

图 4 所提方案时序图

Fig. 4 Sequence diagram of the proposed scheme

4.2 方案具体构造

本小节所使用的符号定义如表1所列。

表1 使用符号说明

Table 1 Usage of symbols

参数	含义
SK_c,SK_s	客户端、MQTT Broker SM2私钥
PK_c,PK_s	客户端、MQTT Broker SM2公钥
$K1$	客户端与MQTT Broker SM4会话密钥
Key	订阅端与发布端之间的SM4会话密钥
$Client_{ID}$	客户端身份
$Broker_{ID}$	MQTT Broker身份
$Topic$	主题
$KeyTopic$	关键密钥组件
$message$	主题的消息内容
$username$	用户名
$password$	密码
H_c	ClientID哈希值
$C1$,$C1'$	身份认证阶段发送的密文
C_{up}	Username||password密文
C_{KT}	KeyTopic密文
C_{mess}	主题的消息内容密文
$\|\|$	连接操作
σ_1,σ_2	C1′的签名
$E()$,$D()$	SM4算法加密、SM4算法解密
$H()$	SM3哈希算法
$Sig_{SK}()$	SM2算法私钥签名
$Ver_{PK}()$	SM2算法公钥验签

4.2.1 初始化阶段

该阶段由KGC执行。首先,KGC选取SM2算法规定的椭圆曲线 $E_p(a,b)$。然后,KGC在 $Ep(a,b)$ 上选择基点 $G(x_G,y_G)$ 以及基点 G 的阶数 n,并选择基点的余因子 h。最后,KGC选择一个安全哈希函数 $H:\{0,1\}^* \rightarrow \{0,1\}^n$。KGC输出的系统公开参数为 $param=(p,E_p(a,b),G,n,h,H)$。

4.2.2 密钥生成阶段

该阶段由客户端和MQTT Broker执行。算法输入公开参数 $param$,输出公私钥对。首先,实体随机选取 $d\in[1,n-2]$ 作为私钥。然后,计算椭圆曲线上的点 $P=(x_p,y_p)=dG$ 作为公钥。最后,算法输出SM2密钥对 (d,P),并公开自身公钥 P。

4.2.3 身份认证阶段

在此阶段,客户端与MQTT Broker可以进行双向的身份认证。为了提升安全性,MQTT Broker通过本地身份列表与SM2数字签名的方式完成第一次身份认证后,再通过MQTT Broker传统的用户名和密码完成认证。值得注意的是,本阶段的用户名和密码是以密文的形式传输,因此攻击者无法获得其明文。身份认证具体步骤如下:

1)客户端在其私钥 SK_c 中随机选出一组数 $K1\in Z_2^{128}$,通过MQTT Broker的公钥 PK_s 对 $K1$、自身 $Client_{ID}$ 进行加密(SM2算法)生成 $C1$,将客户端公钥 PK_c 和 $C1$ 发送至MQTT Broker。

2)MQTT Broker通过自身私钥 SK_s 对 $C1$ 进行解密,获得客户端信息。将获得的 $Client_{ID}$ 与本地身份列表进行对比,若身份列表中有相同的 $Client_{ID}$,则计算

$$H_c=H(Client_{ID}) \tag{3}$$

将 H_c 和 $Client_{ID}$ 以及 $K1$ 对应的身份存放在身份列表中。令 $C1=C1'$,用自身私钥 SK_s 对 $C1'$ 进行签名,即

$$\sigma 1=Sig_{SK_s}(C1') \tag{4}$$

将 $C1'$ 和 σ_1 发送给客户端。

3)客户端将接收到的 $C1'$ 与本地 $C1$ 进行比对,如果完全相同,则用本地存放的MQTT Broker公钥验签,即 $Ver_{PK_s}(\sigma_1)$。在签名验证通过后,客户端用 SK_c 对 $C1'$ 进行签名,即

$$\sigma_2=Sig_{SK_c}(C1') \tag{5}$$

将 $C1'$ 和 σ_2 发送给MQTT Broker。

4)MQTT Broker将接收到的 $C1'$ 和本地 $C1$ 进行比对,如果完全相同,则执行 $Ver_{PK_c}(\sigma_2)$。在签名验证通过后返回验证结果,$K1$ 为客户端与MQTT Broker之间的SM4会话密钥。

5)客户端通过 $K1$ 对 $username$ 和 $password$ 进行加密,即

$$C_{up}=E_{K1}(username\ ||\ password) \tag{6}$$

计算 H_c 并将其和 C_{up} 放入CONNECT数据包中的对应位置后发送至MQTT Broker。

6)MQTT Broker收到CONNECT数据包后通过 H_c 找到对应 $Client_{ID}$ 的SM4密钥 $K1$,用 $K1$ 解密密文 C_{up} 得到 $username$ 和 $password$,即

$$username\ ||\ password=D_{K1}(C_{up}) \tag{7}$$

在身份验证通过后返回CONNACK给客户端。

4.2.4 关键密钥组件获取阶段

在关键密钥组件获取阶段,发布端与订阅端获取到相同的关键密钥组件,用于组装相同的SM4会话密钥。为了增强密钥的强度,本文所提方案中,客户端和MQTT Broker共同参与关键密钥组件的制作。关键密钥组件获取的具体步骤如下:

1)在订阅端SUBSCRIBE数据包和发布端PUBLISH数据包发送前,将 H_c 和订阅主题 $Topic$ 发送至MQTT Broker。

2)MQTT Broker查看本地是否有相同的 $Topic$。如果没有相同的 $Topic$,则MQTT Broker计算

$$KeyTopic=H(Topic\ ||\ K1\ ||\ SK_s) \tag{8}$$

并将 $KeyTopic$ 与 $Topic$ 存放到本地。如果有相同的 $Topic$,则直接通过 $Topic$ 找到对应的 $KeyTopic$。

3)MQTT Broker通过客户端(订阅端、发布端)发送的 H_c 在本地身份列表中找到对应的SM4密钥 $K1$,用 $K1$ 加密 $KeyTopic$,即

$$C_{KT}=E_{K1}(KeyTopic) \tag{9}$$

并将 C_{KT} 发送至客户端。

4)订阅端接收到密文 C_{KT} 并解密,即

$$KeyTopic=D_{K1}(C_{KT}) \tag{10}$$

4.2.5 数据传输阶段

在获取到关键密钥组件后即可进行订阅主题与发送主题的消息内容传输。值得一提的是,在本阶段,除了通过SM4算法对数据进行加密与解密过程外,还进行了以下两点操作:

1)传输密钥组件。在上一阶段,发布端与订阅端已经获得了相同的关键密钥组件,而这一阶段在发布端加密并发送主题的消息内容的同时,还发送了组成密钥的另一密钥组件,且本文所提方案通过SM3哈希算法组装密钥,时间开销可以忽略。

2)通过SM3算法对传输的数据进行完整性检测,既保证了密钥组件的准确性,也保证了主题的消息内容的完整性。

数据传输阶段具体步骤如下：

1）订阅端发送 SUBSCRIBE 数据包至 MQTT Broker，订阅主题。

2）发布端生产随机数 $R\in Z_2^{128}$，并计算

$$H_{\mathrm{Key}}=H(Topic\parallel R\parallel KeyTopic) \tag{11}$$

选取 H_{Key}的前 16 个字节作为数据加密密钥 Key，对主题的消息内容进行加密，即

$$C_{\mathrm{mess}}=E_{\mathrm{Key}}(message) \tag{12}$$

发布端发送至 MQTT Broker 的 PUBLISH 数据包的 Payload 中的内容为

$$Payload=C_{\mathrm{mess}}\parallel R\parallel H(C_{\mathrm{mess}}\parallel R\parallel KeyTopic) \tag{13}$$

3）订阅端接收到 MQTT Broker 转发的 PUBLISH 数据包后提取 Payload 中的内容并计算

$$H(C_{\mathrm{mess}}\parallel R\parallel KeyTopic) \tag{14}$$

将其与接收到的 $H(C_{\mathrm{mess}}\parallel R\parallel KeyTopic)$进行对比，如果相同，则计算

$$H_{\mathrm{Key}}=H(Topic\parallel R\parallel KeyTopic) \tag{15}$$

并选取 H_{Key}前 16 个字节作为解密密钥 Key 解密消息 C_{mess}。至此得到主题的消息内容明文。

5 安全性分析

5.1 身份认证阶段安全性分析

1）假设 Attacker 冒充客户端与 MQTT Broker 进行连接，由于 MQTT Broker 中配置了身份列表，该列表中没有 Attacker 的 ID，因此 Attacker 无法连接 MQTT Broker。

2）假设 Attacker 通过中间人攻击获取到密文 $C1$ 和客户端公钥 PK_c，由于 Attacker 无法得知 MQTT Broker 的私钥，因此无法对密文进行解密，进而无法获取密钥 $K1$ 和 $Client_{\mathrm{ID}}$。Attacker 想要对数据进行非法篡改包括以下 3 种情况：

（1）如果 Attacker 篡改密文数据，则 MQTT Broker 在接收到密文后通过自身私钥 SKs 解密会失败报错。

（2）如果 Attacker 对公钥 PK_c 进行篡改，则接下来 MQTT Broker 通过验签时就会验证失败。

（3）如果 Attacker 截获到客户端向 MQTT Broker 发送的第一条消息并将 PK_c 替换成自己的公钥，那么在 MQTT Broker 验签时还是会验证失败。因为 SM2 算法进行签名验签操作时需要用到 Z_{A}，在预备知识中提到 Z_{A} 是关于用户 A 的可辨别标识、部分椭圆曲线系统参数和用户 A 公钥的杂凑值，尽管 Attacker 可以保证椭圆曲线系统参数和公钥的正确性，但是在本文所提方案中，$Client_{\mathrm{ID}}$ 一直是以密文传输，Attacker 无法获取到 $Client_{\mathrm{ID}}$。

3）假设 Attacker 通过某种方式获取到 $Client_{\mathrm{ID}}$并顺利通过了第一次身份认证，由于本文所提方案将用户名与密码通过 SM4 算法进行加密保护，因此 Attacker 无法获取到用户名与密码，也就无法通过第二次身份认证。

5.2 数据传输阶段安全性分析

1）在获取关键密钥组件的过程中，$Client_{\mathrm{ID}}$是以哈希值传输的。故 Attacker 无法在此阶段获取到 $Client_{\mathrm{ID}}$。

2）假设 Attacker 通过中间人攻击获取到传输的 C_{KT}，但由于 Attacker 没有会话密钥 $K1$，因此无法解密 C_{KT}进而获得 KeyTopic。

3）假设 Attacker 在之前的会话中截获了某一 $Client_{\mathrm{ID}}$的哈希值，并以此哈希值冒充客户端想要获取关键密钥组件。尽管其可以接收到 $KeyTopic$ 的密文 C_{KT}，但同样也会因为没有会话密钥 $K1$，而无法解密 C_{KT}。

4）数据传输阶段通过 SM3 算法对传输的数据计算哈希值。假设 Attacker 截获并修改主题的消息内容密文 C_{mess} 或者密钥组件 R，当订阅端收到 Payload 的内容后用相同的算法计算哈希值，若两个哈希值不相等则表示数据不完整或被非法篡改。

5）在数据传输阶段用到的主题的消息内容加密密钥 Key 是由 $KeyTopic$ 和 R 组成的，其中 R 以明文传输，但是通过上面的分析可知 Attacker 无法获取 $KeyTopic$，所以密钥 Key 是安全的。

5.3 安全性对比

为了彰显本文所提方案的安全性，本小节将其与其他的 MQTT 协议安全方案进行比较，具体如表 2 所列。

表 2 方案安全性对比

Table 2 Comparison of security with related works

方案	单向身份认证	双向身份认证	端到端加密
文献[10]方案	×	×	√
文献[12]方案	√	√	×
文献[14]方案	√	√	×
文献[19]方案	×	×	√
本文所提方案	√	√	√

其中，单向身份认证指只有 MQTT Broker 对客户端的认证；双向身份认证指 MQTT Broker 和客户端互相认证；端到端加密指主题的消息内容可以在订阅端解密获得，MQTT Broker 只对主题的消息内容做转发处理而无须解密消息。

6 仿真实验

本章从计算开销方面对本文所提方案与其他现有方案做了比较。从 CPU 占用率方面对本文所提方案与传统的 MQTT 协议做比较，实验结果表明，本文所提方案在保证 MQTT 协议安全的前提下，增加的开销和 CPU 占用率完全可以满足实际的应用需求。仿真实验环境为：Intel(R) Core (TM) i5-12500H@3.1 GHz CPU，16GB RAM，Windows11 64 位操作系统。

6.1 计算开销

本文所提方案分别采用 SM2 算法和传统 MQTT 协议的用户名密码两个阶段对连接 MQTT Broker 的客户端进行身份认证，因此，本实验分别统计了身份认证的两个阶段的计算开销。图 5 给出了身份认证第一阶段计算开销，实验结果表明，身份认证第一阶段的平均计算开销为 17.3 ms。

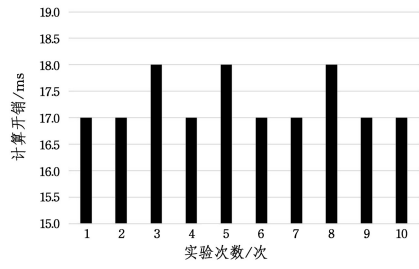

图 5 身份认证第一阶段计算开销

Fig. 5 Computation cost of identity authentication in first stage

身份认证第二阶段是通过传统的 MQTT 协议存在的用户名与密码进行的。由于传统 MQTT 协议的用户名和密码是以明文传输，因此其身份认证机制比较脆弱。本文所提方案通过 SM4 算法对用户名和密码进行加密。实验统计了第二次身份认证增加的计算开销。图 6 给出了身份认证第二阶段计算开销，实验结果表明，本文所提方案的身份认证第二阶段的平均计算开销为 3 μs。

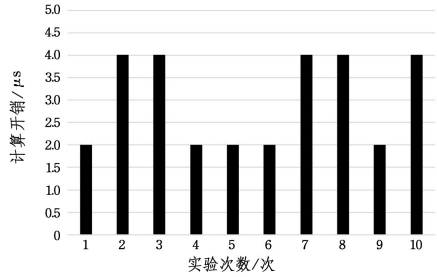

图 6 身份认证第二阶段计算开销

Fig. 6 Computation cost of identity authentication in second stage

关键密钥组件获取阶段是本文所提方案增加的阶段，实验统计了关键密钥组件获取阶段增加的计算开销。如图 7 所示，实验结果表明，该阶段的平均计算开销为 57.4 μs。

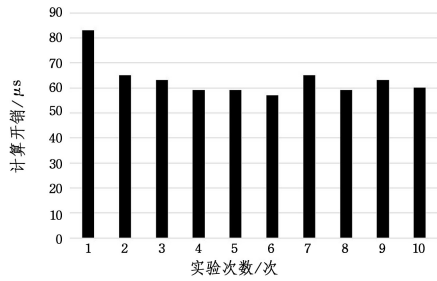

图 7 密钥组件获取阶段计算开销

Fig. 7 Computation cost of key component acquisition

在数据传输阶段，实验统计了增加的计算开销，其中包括发布端和订阅端的数据加密算法与哈希算法的使用开销。如图 8 所示，实验结果表明，该阶段的平均计算开销为 945.6 μs。

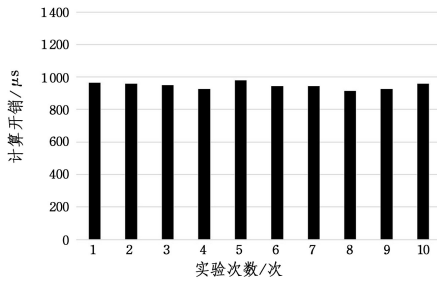

图 8 数据传输阶段计算开销

Fig. 8 Computation cost of data transmission

综上，本文所提方案身份认证、关键密钥组件获取、数据传输 3 个阶段的总平均计算开销为 18.306 ms。其中，大部分计算开销来自于身份认证阶段。

计算开销的对比如表 3 所列，其中 E 表示指数运算，H 表示哈希运算，SSL 表示使用一次 SSL 协议，AES 表示一次 AES 算法加解密运算，SM4 表示一次 SM4 算法加解密运算，SM2S 表示一次 SM2 算法签名验签运算，SM2E 表示一次 SM2 算法加解密运算，MAC 表示消息认证码运算。为了使实验结果更加严谨直观，实验中分别统计了 KP-ABE 算法、指数运算、SSL 协议、AES 算法、哈希算法、MAC 算法的计算开销。其中，KP-ABE 算法在一个属性时加解密的计算开销约为 85 ms，一次指数运算的计算开销约为 8 ms，一次 SSL 协议握手的开销约为 150 ms；而 AES 算法、哈希算法、MAC 算法的计算开销均不到 1 ms。

表 3 不同方案计算开销对比

Table 3 Comparison of computation cost with related works

方案	计算开销
文献[10]方案	KP-ABE +AES
文献[12]方案	2AES+4MAC+6E+8H
文献[14]方案	SSL+6H
文献[19]方案	AES+14E+4H
本文所提方案	3SM4+SM2E+2SM2S+7H

如表 3 所列，文献[10]方案采用的 KP-ABE 算法是一种复杂的公钥加密算法，其加解密的计算代价与属性个数成正比关系，开销远远超过 AES，SM4 等对称加密算法和哈希算法。文献[12]方案多次运用了指数运算，并且该方案没有实现端到端的加密，故其数据机密性较差。文献[14]方案中部分通信采用了 SSL 安全协议，SSL 安全协议中涉及的数字证书的管理和验证所带来的计算开销和性能损耗远远超过 SM2，SM4，SM3 算法。文献[19]方案采用了大量的指数运算，其计算开销较大，并且该方案没有实现客户端与 MQTT Broker 之间的身份认证。

综上所述，与现有的安全方案相比，本文所提方案在增加了安全功能的基础上所增加的计算开销较小且优于其他安全方案。

6.2 CPU 占用率

此次实验以 200 bytes，400 bytes，600 bytes，800 bytes，1 000 bytes 作为数据传输阶段的测试数据大小，对 CPU 占用率进行了对比。将实验中获取的数据归纳为两个表和一个图。首先实验测得传统 MQTT 协议从客户端连接 MQTT Broker 到订阅端接收到主题的消息内容这一时间段的 CPU 占用率，数据如表 4 所列。其次，通过实验测得本文所提方案在同一时间段的 MQTT 协议 CPU 占用率，数据如表 5 所列。

表 4 传统 MQTT 协议 CPU 占用率

Table 4 CPU utilization of traditional MQTT protocol

(%)

实验次数	200 bytes	400 bytes	600 bytes	800 bytes	1 000 bytes
1	11	13	13	14	14
2	11	12	13	13	14
3	13	12	13	13	13
4	13	13	12	14	14
5	12	13	13	13	13
6	13	13	14	13	14
7	13	12	13	14	14
8	11	13	12	13	13
9	13	12	13	13	14
10	13	12	13	13	14
平均 CPU 占用率	12.3	12.5	12.9	13.3	13.7

表 5 本文所提方案 CPU 占用率

Table 5 CPU utilization of the proposed scheme

(%)

实验次数	200 bytes	400 bytes	600 bytes	800 bytes	1 000 bytes
1	14	17	16	19	23
2	16	20	20	19	22
3	18	14	18	18	21
4	18	15	15	19	22
5	15	14	16	17	17
6	14	15	17	20	20
7	18	20	17	19	16
8	18	19	17	17	22
9	15	17	18	19	19
10	16	17	19	20	19
平均 CPU 占用率	16.2	16.8	17.3	18.7	20.1

最后将两个平均 CPU 占用率进行对比，结果如图 9 所示。

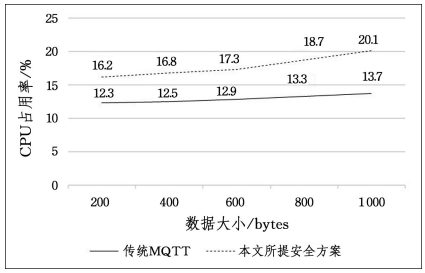

图 9 CPU 占用率对比

Fig. 9 Comparison of CPU utilizations

由图 9 可知，本文所提方案在传统 MQTT 协议的基础上增加了身份认证、数据加密等安全防护措施，故其 CPU 占用率相比传统 MQTT 协议略有提升。如图 9 所示，数据量大小为 200 bytes 时，CPU 占用率提升了 3.9%；数据量大小为 400 bytes 时，CPU 占用率提升了 4.3%；数据量大小为 600 bytes 时，CPU 占用率提升了 4.2%；数据量大小为 800 bytes 时，CPU 占用率提升了 5.4%；数据量大小为 1 000 bytes 时，CPU 占用率提升了 6.4%。根据实验分析可知，本文所提方案在增强了 MQTT 协议安全防护能力的同时，对 CPU 占用率的影响较小。

结束语 本文所提方案采用国密算法 SM2，SM3，SM4 对传统 MQTT 协议进行保护，使其具备双向身份认证与数据加密功能。前者保证了连接 MQTT Broker 的客户端的合法身份，防止攻击者非法连接 MQTT Broker 并传输和获取关键数据信息；后者实现了端到端的数据安全传输，防止攻击者通过中间人攻击非法截获和篡改数据。实验结果表明，本文所提方案在保证 MQTT 协议安全性的同时，还可满足实际的应用需求。由于所提方案在身份认证阶段的通信轮数较多，如何设计实现一个更高效的身份认证，将是下一步研究工作的重点。

参 考 文 献

[1] MOUSTAFA N, TURNBULL B, CHOO K K R. An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things[J]. IEEE Internet of Things Journal, 2018, 6(3): 4815-4830.

[2] WOOD D, APTHORPE N, FEAMSTER N. Cleartext Data Transmissions in Consumer Iot Medical Devices[C]// Proceedings of the 2017 Workshop on Internet of Things Security and Privacy. 2017: 7-12.

[3] AZROUR M, MABROUKI J, GUEZZAZ A, et al. Internet of Things security: challenges and key issues[J]. Security and Communication Networks, 2021, 2021: 1-11.

[4] MILEVA A, VELINOV A, HARTMANN L, et al. Comprehensive analysis of MQTT 5.0 susceptibility to network covert channels[J]. Computers & Security, 2021, 104: 102207.

[5] NAIK N. Choice of Effective Messaging Protocols for IoT Systems: MQTT, CoAP, AMQP and HTTP[C]// 2017 IEEE International Systems Engineering Symposium (ISSE). IEEE, 2017: 1-7.

[6] DINCULEANĂ D, CHENG X. Vulnerabilities and limitations of MQTT protocol used between IoT devices [J]. Applied Sciences, 2019, 9(5): 848.

[7] ZHANG L, GE Y. Identity Authentication Based on Domestic Commercial Cryptography with Blockchain in the Heterogeneous Alliance Network[C]// 2021 IEEE International Conference on Consumer Electronics and Computer Engineering (ICCECE). IEEE, 2021: 191-195.

[8] SHEN C X, GONG B. The innovation of trusted cmputing based on the domestic cryptography[J]. Journal of Cryptologic Research, 2015, 2(5): 381-389.

[9] SINGH M, RAJAN M A, SHIVRAJ V L, et al. Secure mqtt for Internet of Things (iot)[C]// 2015 fifth International Conference on Communication Systems and Network Technologies. IEEE, 2015: 746-751.

[10] BISNE L, PARMAR M. Composite Secure MQTT for Internet of Things Using ABE and Dynamic S-box AES[C]// 2017 Innovations in Power and Advanced Computing Technologies (IPACT). IEEE, 2017: 1-5.

[11] BHAWIYUGA A, DATA M, WARDA A. Architectural Design of Token Based Authentication of MQTT protocol in Constrained IoT device[C]// 2017 11th International Conference on Telecommunication Systems Services and Applications (TSSA). IEEE, 2017: 1-4.

[12] CALABRETTA M, PECORI R, VELTRI L. A Token-based Protocol for Securing MQTT Communications[C]// 2018 26th International Conference on Software, Telecommunications and Computer Networks (SoftCOM). IEEE, 2018: 1-6.

[13] SU W T, CHEN W C, CHEN C C. An Extensible and Transparent Thing-to-thing Security Enhancement for mqtt Protocol in Iot Environment[C]// 2019 Global IoT Summit (GIoTS). IEEE, 2019: 1-4.

[14] CHIEN H Y, CHEN Y J, QIU G H, et al. A MQTT-API-compatible IoT security-enhanced platform[J]. Int. J. Sens. Networks, 2020, 32(1): 54-68.

[15] DE RANGO F, POTRINO G, TROPEA M, et al. Energy-aware

dynamic Internet of Things security system based on Elliptic Curve Cryptography and Message Queue Telemetry Transport protocol for mitigating replay attacks[J]. Pervasive and Mobile Computing, 2020, 61: 101105.

[16] SANJUAN E B, CARDIEL I A, CERRADA J A, et al. Message queuing telemetry transport (MQTT) security: a cryptographic smart card approach[J]. IEEE Access, 2020, 8: 115051-115062.

[17] PATEL C, DOSHI N. A novel MQTT security framework in generic IoT model[J]. Procedia Computer Science, 2020, 171: 1399-1408.

[18] AMANLOU S, BAKAR K A A. Lightweight security mechanism over MQTT protocol for IoT devices[J]. International Journal of Advanced Computer Science and Applications, 2020, 11(7): 202-207.

[19] GU Z C, GUO Y B, FANG C. End-to-end security solution for message queue telemetry transport protocol based on proxy re-encryption[J]. Journal of Computer Applications, 2021, 41(5): 1378-1385.

[20] SPINA M G, DE RANGO F, MAROTTA G M. Lightweight Dynamic Topic-centric End-to-end Security Mechanism for MQTT[C]// 2021 IEEE/ACM 25th International Symposium on Distributed Simulation and Real Time Applications (DS-RT). IEEE, 2021: 1-7.

[21] MENDOZA-CARDENAS F, LEON-AGUILAR R S, QUIROZ-ARROYO J L. CP-ABE Encryption over MQTT for an IoT System with Raspberry Pi[C]// 2022 56th Annual Conference on Information Sciences and Systems (CISS). IEEE, 2022: 236-239.

[22] ZHANG Y, HE D, ZHANG M, et al. A provable-secure and practical two-party distributed signing protocol for SM2 signature algorithm[J]. Frontiers of Computer Science, 2020, 14(3): 1-14.

[23] ABED S, JAFFAL R, MOHD B J, et al. Performance evaluation of the SM4 cipher based on field-programmable gate array implementation[J]. IET Circuits, Devices & Systems, 2021, 15(2): 121-135.

[24] TOLDINAS J, LOZINSKIS B, BARANAUSKAS E, et al. MQTT Quality of Service VersusEnergy Consumption[C]// 2019 23rd International Conference Electronics. IEEE, 2019: 1-4.

[25] AL ENANY M O, HARB H M, ATTIYA G. A Comparative Analysis of MQTT and IoT Application Protocols[C]// 2021 International Conference on Electronic Engineering (ICEEM). IEEE, 2021: 1-6.

LIU Zechao, born in 1985, Ph.D, associate professor, master supervisor. His main research interests include industrial information security and cryptography.

LI Jun, born in 1986, Ph.D. His main research interests include industrial Internet security and data security.

(责任编辑:何杨)