iOS 端基于分组加密算法的密码键盘安全中间件

彭珊珊 [1]
PENG Shanshan

摘 要 设计了一种能独立应用于 iOS 端应用开发中的加密密码键盘安全中间件，支持应用最为广泛的分组加密算法 DES、3DES 及国密算法 SM4，同时考虑到 iOS 未来开发过程中 UI 的适配问题及动设备原生的软键盘布局固定带来的安全等级低的问题，在设计中加入动态 UI 布局，使得每次出现的密码键盘布局不同。设计共分为三个核心模块，分别是加密模块、解密模块、UI 动态布局模块，其中解密模块可结合应用场景单独使用。支持多种加密方式的密码键盘安全中间件，保证了用户在输入隐私数据时的隐私安全，同时提高了项目开发中的维护效率。

关键词 安全中间件；加密密码键盘；信息安全；iOS 系统

doi：10.3969/j.issn.1672-9528.2024.07.032

0 引言

在银行等金融和移动支付领域，通常使用键盘进行支付密码、支付金额、付款人信息等的输入。移动端软件在实际支付场景下，通常使用移动端设备自带的软键盘作为密码键盘，移动设备原生的软键盘布局固定，无法随机动态布局，用户输入的密码可能存在未加密保护的可能，进一步增加用户隐私信息泄露的风险。此外，银行等各种金融及移动支付的应用产品，都需要分别维护不同的密码键盘项目源码，增加维护成本。现提出一种基于 iOS 移动平台的密码键盘安全中间件，该中间件支持布局的动态变化，支持 DES、3DES、国密 SM4 加密算法，保证用户隐私信息的安全。同时，该安全空间可独立集成到各种 iOS 应用中，兼容当前 iOS 平台支持的 Object-C、swift 语言接口。

1 加密算法理论

1.1 DES 算法

DES 算法是分组加密算法，对于明文编码后的二进制序列，分组密码会将其划分成长度固定的组（块），每组分别在密钥的控制下转换成等长的密文分组。分组密码算法的安全策略中，用得最多的就是代换 - 置换网络（substitution-permutation network）），简称 S-P 网络，是由 S 变化（代换）和 P 变化（置换或换位）交替进行多次迭代而形成的变换网络，通过代换和置换（S-P 网络）交替的方式来构造分组密码。该结构是可逆的，也可以用来解密 [1]。DES 算法的初始密钥共 64 位，但在实际加密过程中，密钥的有效位数为 56 位，剩余 8 位是作为奇偶校验位；Data 的分组长度是 8 个字节，共 64 位。

DES 算法的 64 位分组密文输出的过程为：（1）初始置换：初始置换 IP 处理；（2）生成子密钥：子密钥的生成大致经历三个过程，分别是置换选择、循环左移、置换选择，生成的子密钥用于加密密文；（3）迭代过程：初始置换 IP 对明文的 64 位进行换位处理，然后通过子密钥对明文进行 16 轮乘积变换；（4）逆置换：对最终的加密结果进行逆置换，得到最终的 64 位密文输出 [2]。

迭代过程中每一轮采用 Feistel 密码结构，基本思想是先将明文分成两半各 32 位数据，对其中一半进行一系列的加密操作，最后交换两半位置，重复这个过程多轮后得到最终密文 [3]。下面是给出某 64 位的明文经过初始置换后，利用产生的工作密钥进行 16 次迭代的核心流程以及 i-1 次迭代之后输出的明文导入到第 i 次迭代之后的处理流程，得到最终的密文。每次迭代都会用到由初始密钥生成子密钥来加密明文分组，图 1 中从左到右依次是子密钥生成过程，第 i 次迭代过程具体细节见图 2。

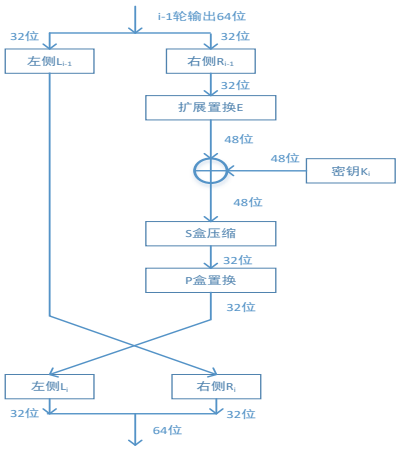

图 1 子密钥生成

1. 北京学而思教育科技有限公司 北京 100085

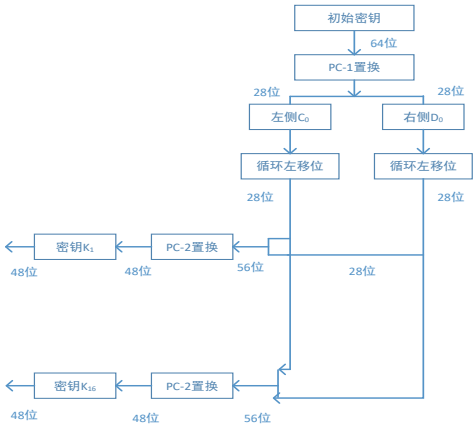

图 2 第 i 次迭代

1.2 3DES 算法

3DES 是三重 DES 方案，有四种使用模式：（1）DES-EEE3 模式，使用三个不同密钥（k_1, k_2, k_3），采用三次加密算法；（2）DES-EDE3 模式，使用三个不同的密钥（k_1, k_2, k_3），采用加密 - 解密 - 加密算法；（3）DES-EEE2 模式，使用两个不同密钥（k_1=k_3, k_2），采用三次加密算法；（4）DES-EDE2 模式，使用两个不同密钥（k_1=k_3, k_2），采用加密 - 解密 - 加密算法 [4]。其中，前两种加密方式总密钥长度达到 56×3，共 168 位，奇偶校验位 3×8，共 24 位，后两种密钥长度达到 56×2，共 112 位，能够更加有效地克服 DES 算法的穷举搜索危机，安全性能更高。

1.3 国密 SM4 算法

SM4 是我国国家密码管理局在 2012 年发布的分组加密算法，其分组长度为 128 位，密钥长度也为 128 位，其加密过程采用 32 轮迭代机制。这个迭代机制与 DES 加密算法类似，每一轮需要一个轮密钥，此处的轮密钥类似于 DES 加密算法中的子密钥 [5]。

此处说明一个字就相当于 32 位，SM4 加密的分组长度相等于 4 个字。当输入 4 个字的明文（X_0, X_1, X_2, X_3）时，经过加密之后，得到输出的 4 个字的密文（Y_0, Y_1, Y_2, Y_3），之后进行一次反序变换，得到最终的密文 [6]。SM4 的解密过程与加密过程类似，也是经过 32 轮迭代及 1 次反序变换完成，但是解密过程的轮密钥使用的是加密中轮密钥的逆序 [7-8]。

2 动态密码键盘的应用场景

iOS 端应用开发时，加密和解密在同一端，可同时在网络端本地端进行加密和解密。加密和解密分开进行，本地端加密、网络端解密或者网络端加密、本地端解密，保证密码以密文形式传输的安全性。

3 密码键盘安全中间件设计

本文设计的基于 iOS 移动端的动态密码键盘分为三个模块，分别是加密模块、解密模块及 UI 布局模块。加密模块及解密模块支持 DES、3DES、SM4 加密算法，其中算法模块利用 polarssl 中的算法模块实现密钥设置 [9-10]、加密及解密的过程，可支持的最大密码长度为 1024 位；UI 布局支持字母、数字、符号键盘的单独动态布局及混合动态布局。

各模块中使用公共的宏定义，标志加密及解密算法的类型，此类型可继续后续其他加密算法的扩展使用。

```
typedef enum
{
    CUSTOM_ ENCWAY_ DES,
    CUSTOM_ ENCWAY_ 3DES,
    CUSTOM_ ENCWAY_SM4,
    ……
}CUSTOM_ ENCWAY;
```

各模块中使用公用的错误码标识，用于确定加密、解密及 UI 变换过程中的错误问题。

```
typedef enum
{
CUSTOM_ KEYBOARD_ 0K;    /* 成功 */
CUSTOM_ KEYBOARD_ ERR;    /* 密钥错误 */
CUSTOM_ KEYBOARD__OP_ DISABLE; /* 操作禁止 */
CUSTOM_ KEYBOARD_ MAXLEN;  /* 密码长度错误 */
}CUSTOM_ KEYBOARD_ RV;
```

UI 布局模块中使用公用的键盘 UI 类型标识，用于确定当前需要的键盘 UI。

```
typedef enum
{
    CUSTOM_ KEYBOARD_AlphaetView;
    CUSTOM_ KEYBOARD_ NumberView;
    CUSTOM_ KEYBOARD_ SymbolView; .
}CUSTOM_ KEYBOARD_ STYLE;
```

3.1 加密模块

加密模块核心有两个接口，分别是设置加密密钥接口及获取密文接口。设置密钥接口用于对密码键盘输入数据进行数据加密过程中密钥的设置，获取密文接口用于实时获得加密后的密文。

setEncKeyAndWay 接口共有三个参数，分别是密钥数据、密钥长度及密钥类型，通过返回错误码确定密钥是否设置成功。

```
CUSTOM KEYBOARD_ RV setEncKeyAndWay(NSData*) encKey encWay: (CUSTOM ENCWAY)encWay
{
    if(kDidCreatedKeyboard)
      return CUSTOM_ KEYBOARD_ OP_ DISABLE;
    kEncKey = encKey;
    ……
}
```

设置密钥接口是未设置输入密码数据之前进行设置密钥，否则密码键盘使用过程是未加密的情况，设置密钥可在应用程序中重复设置，在密码键盘 UI 未弹出之前设置密钥能够保证密码键盘使用过程中使用设置的密钥对输入的密码数据进行加密。

getText 接口不需要入参，通过先前设置的密钥类型及密钥，获得加密之后的密码数据，此接口对用户屏蔽密钥类型及密钥数据。如果每次获取密文，密文在动态存储空间中的相对位置固定，就很容易被黑客渗透测试发现密文的位置规律。在此接口开始先申请足够大的动态存储空间，通过 iOS 中的真正伪随机数生成算法 arc4random 生成多次随机数，覆盖为存储密文而申请的动态存取空间，接下来使用申请的最大动态空间大小、密文数据的最大长度利用 arc4random 和模运算 % 生成一个随机数，作为密文在动态存储空间中的首位置偏移地址，如下代码，生成一个 0~ENCDATASPACESIZE-MAXCLEATEXTSIZE-9 之间的一个随机数，作为之后密文存储的首位置偏移地址。

unsigned int encDataOffset =
arc4random() % (ENCDATASPACESIZE - MAXCLEATEXTSIZE- 9)

确定密文在动态存储空间中首位置的偏移地址，利用密钥的类型（DES、3DES、SM4）初始化对应的加密算法上下文 context，在 DES、3DES 加密算法中进行 PKCS#5 填充，保证最终密文数据长度填充到 8 的倍数。假设原数据长度为 x，则填充后的长度为 x+(8−(x%8)) 的长度，具体填充的数据为 8−(x%8)，在 SM4 加密算法中参考 PKCS#5 填充方式，将密文数据长度填充到 16 的倍数。假设原数据长度为 x，则填充后的长度为 x+(16−(x%16)) 的长度，具体填充的数据为 16−(x%16)。三种不同的加密算法密文获取方式如下代码，获得密文数据之后，将加密算法上下文释放，密文动态存储空间释放，防止再次调用密文接口使用同一段密文存储空间。

des_setkey_enc(des_context * ctx, const unsigned char key[DES_ KEY_SIZE]);

des_crypt_ecb(des_context * ctx, const unsigned char input[8], unsigned char output[8]);

3des_setkey_enc(3des_context * ctx, const unsigned char key[DES_KEY_SIZE*3]);

3des_crypt_ecb(3des_context * ctx, const unsigned char input[8], unsigned char output[8]);

sm4_setkey_enc(sm4_context * ctx, const unsigned char key[SM4_ KEY _SIZE]);

sm4_crypt_ecb(sm4_context * ctx, const unsigned char input[16], unsigned char output[16]);

3.2 解密模块

解密模块核心接口是对加密的数据进行解密，获得明文的过程。该接口可结合 iOS 应用开发的实际场景进行使用，当需要在应用程序端使用明文时，可用此接口进行数据的解密。解密模块在密码键盘安全空间设计中可与加密模块及 UI 布局模块分离单独使用，在加密端和解密端分离的情况下，解密模块能够单独分离的特点会更兼容此场景。

解密模块通过密钥的类型及密钥对密文数据进行解密获得明文数据。该接口支持 DES、3DES、SM4 三种解密算法，三种算法的解密过程就是加密迭代过程的逆过程，使用由初始密钥获得的子密钥与加密过程使用的子密钥逆序。

在此接口开始先申请足够大的动态存储空间，通过 iOS 中的真正伪随机数生成算法 arc4random 生成多次随机数，覆盖为存储明文而申请的动态存取空间，接下来使用申请的最大动态空间大小、明文数据的最大长度利用 arc4random 和模运算 % 生成一个随机数，作为明文在动态存储空间中的首位置偏移地址，如下代码，生成一个 0~ ENCDATASPACESIZE-MAXCLEATEXTSIZE-9 之间的一个随机数，作为之后明文存储的首位置偏移地址。

unsigned int decDataOffset =arc4random()% (ENCDATASPACESIZE-MAXCLEATEXTSIZE-9);

三种不同的解密算法明文获取方式如下代码，获得明文数据之后，将解密算法上下文释放，明文动态存储空间释放，防止再次调用明文接口使用同一段明文存储空间。

des_setkey_dec(des_context * ctx, const unsigned char key[DES_ KEY_ SIZE]);

des_crypt_ecb(des_context * ctx, const unsigned char input[8], unsigned char output[8]);

3des_ setkey_dec(3des_context * ctx, const unsigned char key[DES_KEY_SIZE * 3]);

3des_crypt _ecb(3des_context * ctx, const unsigned char input[8], unsigned char output[8]);

sm4_ setkey_dec(sm4_context * ctx, const unsigned char key[SM4_KEY_SIZE]);

sm4_crypt_ecb(sm4_context * ctx, const unsigned char input[16], unsigned char output[16]);

3.3 UI 布局模块

UI 布局模块共有三个重要的布局设置思想，分别是字母、数字、符号密码键盘的布局，动态密码键盘变化布局，键盘按键响应事件。

在字母、数字、符号键盘布局中，将密码键盘涉及的字母、数字、符号分别作为独立的字符串，在使用时进行字符串的切割获取到当前按键的显示内容，这种设计也是后续动态密码键盘布局的前提设计。具体按键使用 iOS 中的 UIButton 作为密码键盘中的按键，结合密码键盘中字符串的切割设计，即使在密码键盘错乱的情况下，也无需记住每个按键的 tag 就能识别当前按键的内容。在将每个按键 UIButton 插入当前 UIView 的时候，使用的是行列及与当前 frame 中的宽度、高度结合的形式计算出当前 UIButton 在密码键盘中的位置，此处定义

了一个通用于每个按键的相对位置表示。

```
ENCKEYBOARD_GridRect:
typedef struct{
    int row;
    int column;
    int rowSpan;
    int columnSpan;
}ENCKEYBOARD_ GridRect;
```

根据当前 UIView 的宽度、按键的间隔，能够计算出每个按键的宽度作为字母、数字、符号每个按键的宽度，此处针对 return、空格等特殊按键会按键字母按键宽度的倍数作为其 rowSpan。根据当前 UIView 的高度、按键行间距，能够计算出每个按键的高度作为字母、数字、符号按键的高度，针对 return、空格等特殊按键的高度与字符、数字、字符按键的高度一致。每个按键的 row、column 从 1 开始编号，根据计算的高度、宽度、间隔依次计算每个按键的 row、column、rowSpan、columnSpan， 其 中 的 rowSpan 和 columnSpan 是针对按键当前位置结合宽度、间隔距离依次往前类推。

动态密码键盘变化布局可以实现每次看到的键盘布局与上次相比是打乱的情况，这个设计的关键在于将开始的字母、数字、符号的源字符串进行打乱，即使在打乱顺序的情况下，也依然不会对上述提到的键盘布局有任何影响。本文使用的是字符串切割的形式作为按键的内容，即使源字符串内容顺序错乱，也不会影响后续的切割及布局的设置。

键盘响应事件中，对于字母、数字、特殊符号的内容按键的响应是获得并识别其输入的内容，对于有字母键盘切换数字键盘或符号键盘及大小写切换的响应时间会对对应的 UI 进行切换，完成输入按键的响应事件可以从这里调用 getText 接口，获得密码的密文数据。

需要调起加密密码键盘的输入接口：

```
(CUSTOM_ KEYBOARD_ RV)setTextField: (UlTextField * )textField;
```

隐私数据输入完成的响应事件设置：

```
(CUSTOM_ KEYBOARD_ _RV)setOKActionOnTarget: (id)target 0KAction: (SEL)OKAction;
```

3.4 密码键盘安全中间件三种加密算法测试

在测试该加密密码键盘安全控件过程中，测试设备有 iPhone14、iPhone 14 Plus、iPhone14、iPhone 14 Plus 及 iPad Air5、iPad10 代、iPad9 代，测试的系统有 iOS15、iOS16，测试中着重测试三种加密算法的可用性及 UI 的适配性，待加密的密码使用 NSString 类型，方便 iOS 应用开发，兼容 OC 及 swift，加密数据兼容 polarssl，使用十六进制表示。三种加密算法的可用性通过下面三组测试作为示例。

（1）DES 加密算法

加密密钥：89076543，加密类型： CUSTOM_ENCWAY _DES，待加密的密码：kgcx879@，加密后的密码：0xet145f1 aed986f0be696b7d7d09dcd79，解密后的密码：kgcx879@。

（2）3DES 加密算法

加密密钥：890765432980765423451673，加密类型：CUSTOM_ENCWAY_3DES，待加密的密码：kgcx879@，加密后的密码： 0x3e60e9a43c5e3c660d7e1d774a0c6818，解密后的密码：kgcx879@。

（3）SM4 加密算法

加密密钥：8907654329807654，加密类型：CUSTOM_ ENCWAY _SM4，待加密的密码：kgcx879@，加密后的密码：0x0f413569eb66b50be0c1db9c518fb909，解密后的密码：kgcx879@。

4 未来展望

该加密密码键盘安全中间件能够独立集成到 iOS 应用开发中，独立维护，实现隐私数据的保密性，为用户输入隐私数据提供安全保证。未来结合项目需求，可持续扩展支持其他加密算法和方式，例如 RSA 非对称加密及同态加密等，保证隐私数据安全输入的情况下，又能确认用户身份，为隐私数据提供更多安全保障。

参考文献：

[1] 余启航 , 李斌勇 , 杨雄凯 , 等 .DES 加密算法的过程分析研究 [J]. 网络安全技术与应用 , 2018(2):43-44.

[2] 石姗 , 王勇 , 周林 . 基于 3DES 和 SM4 的 Modbus 安全通信算法 [J]. 自动化博览 , 2021,38(1):67-71.

[3] 张彦峰 . 基于 3DES 算法的电子锁安全性研究 [D]. 武汉 : 华中科技大学 , 2019.

[4] 朱哲明 , 赵泽茂 , 吕金鹏 . 基于 Java 语言实现手机短信加密 [J]. 保密科学技术 , 2012(4):52-56.

[5] 张雨 . 软硬件协同设计的 SM4 密码算法加速器研究与实现 [D]. 西安 : 西安电子科技大学 , 2022.

[6] 丘敬云 , 吴祥晨 , 周宇坤 , 等 . 国密 SM2、SM4 混合算法在车联网中的应用研究 [J]. 仪器仪表用户 ,2023,30(8):9-12 +99.

[7] 张德强 . 探析使用国密算法进行个人信息保护 [J]. 网络空间安全 , 2023,14(5):22-28+66.

[8] 张玉安 , 漆骏锋 , 王野 , 等 . 分组密码的隐秘密文分组链接模式 [J]. 信息安全与通信保密 , 2022(11):92-99.

[9] 周伟 , 郑世珏 .Open SSL 分组加密的时间侧信道分析 [J]. 信息技术 , 2019(1):6-10.

[10] 贺林声 . 自适应混合图像的防窥视密码键盘研究 [D]. 重庆 : 重庆邮电大学 , 2019.

【作者简介】

彭珊珊（1989—），女，山东青岛人，本科，研究方向：计算机应用技术。

（收稿日期：2024-03-27）