九鼎 JDKEY98 多功能加密锁

开发手册

南通市九鼎软件科技有限公司

www.jdkey.cn

修订记录：

修订日期	版本	修订内容
2017 年 1 月	V1.0	第一版发布

一、产品概述

产品功能

JDKEY98 是一款可方便的用于简单型的身份认证和软件加密领域的多功能、多用途 USBKEY 产品。用作冲击响应身份认证时，支持标准的 HMAC_MD5、HMAC_SHA1、HMAC_SHA256、HMAC_SM3 算法；用作软件保护和数据加密时，支持标准的 TEA-128、TDES-128 算法;另外提供有 2 个 1K 的存储区，8 字节硬件序列号，随机数发生器，LED 灯的控制等功能，可以满足一般化的使用需求

主要技术指标

- 采用智能卡安全芯片，确保绝无硬件复制的可能性
- 有随机数参与的 RSA1024+TDES128 的加密通信，能抵御重放攻击
- 基于匿名、用户、管理员的三级权限管理体系
- 支持多种标准冲击响应身份认证算法：

  HMAC_MD5、HMAC_SHA1、HMAC_SHA256、HMAC_SM3
- 支持常用对称加解密算法：TEA-128、TDES-128
- 带有 2 个 1K 的存储区

  REGION_1 权限为匿名可读、用户以上可写

  REGION_2 权限为用户可读、管理员可写
- 有带 128K 光盘的型号，可作为网站登陆器、安装包加载器、说明书等的载体

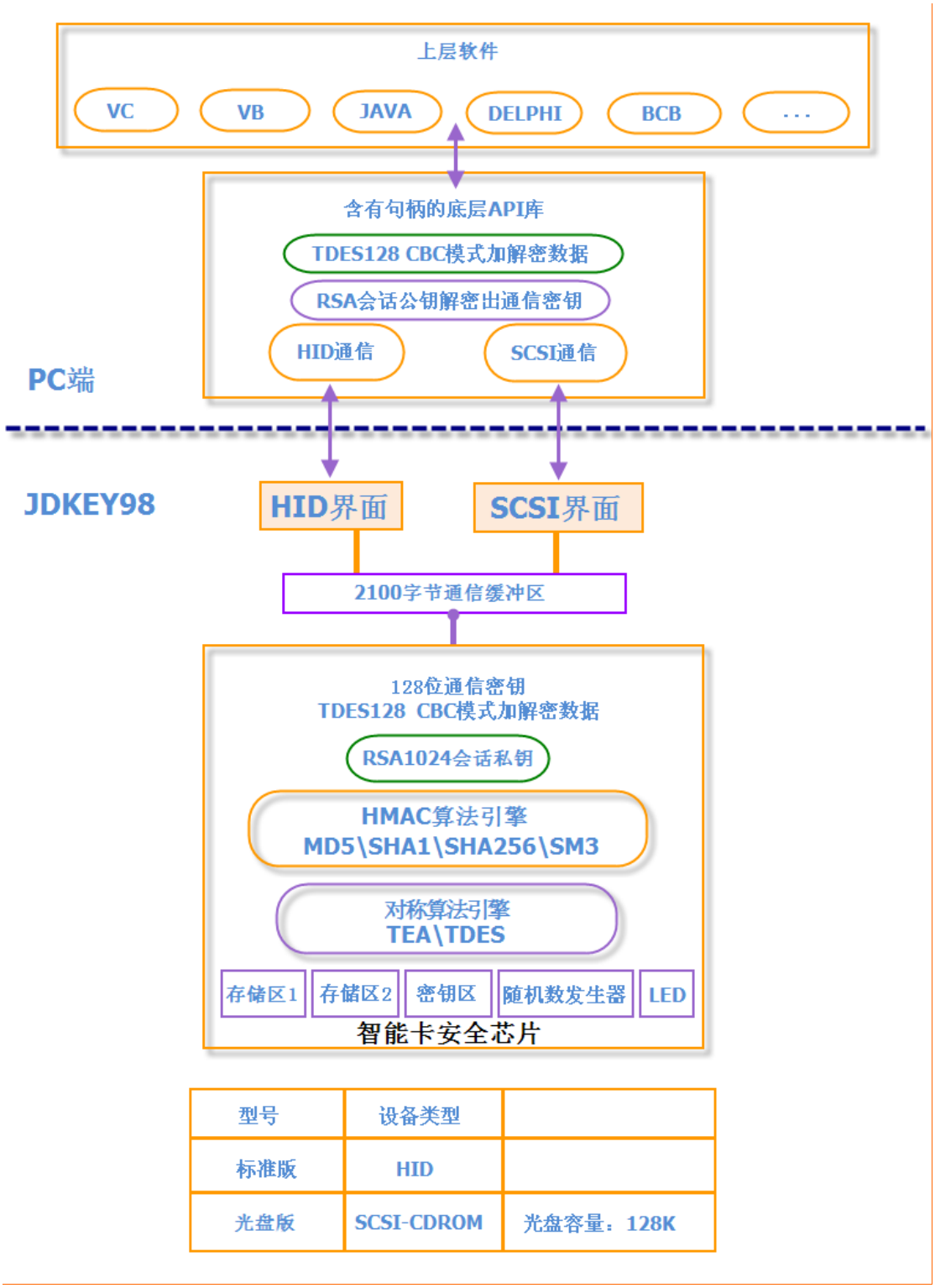
出厂默认设置

用户密码：12345678

管理员密码：FFFFFFFFFFFFFFFF

用户密码和管理员密码错误时的允许重试次数均为 255，即无限制

系统结构框图



型号	设备类型	
标准版	HID	
光盘版	SCSI-CDROM	光盘容量：128K

二、应用程序接口 API

1. JDKey_Enum 枚举加密锁

*DWORD WINAPI JDKey_Enum(OUT DONGLE_INFO * pDongleInfo, IN OUT int * pCount);*

说明：本函数用于枚举出计算机上所有的 JDKEY98 设备

参数：

pDongleInfo [out]: 接收设备信息的数组,此参数可以为 NULL

pCount [in][out]: 输入需要找的最大设备数目,0 表示不限制,返回实际找到的数目

返回值：

DONGLE_SUCCESS 执行成功

2.JDKey_Find 查找指定加密锁

*DWORD WINAPI JDKey_Find(IN char * pProductID, OUT int * pCount);*

说明：本函数用于查找计算机上指定产品 ID 加密锁的个数

参数：

pProductID [in]: 指示要查找的产品 ID

pCount [out]: 返回找到的锁的数目

返回值：

DONGLE_SUCCESS 执行成功

3. JDKey_Open 打开加密锁

*DWORD WINAPI JDKey_Open(OUT DONGLE_HANDLE * phDongle, IN char * pProductID, IN int nIndex);*

说明：本函数用于打开指定的加密锁(可以不需要先 Enum 或 Find)

参数：

phDongle [out]: 设备句柄指针,如果打开成功,会被填充设备句柄

pProductID [in]: 指向 ANSI 的 8 字节产品 ID 字符串

index [in]: 基于 0 的索引值,指示打开找到的第几把加密锁

返回值：

DONGLE_SUCCESS 执行成功

DONGLE_NOT_FOUND 未找到指定的加密机

4.JDKey_Close 关闭加密锁

DWORD WINAPI JDKey_Close(IN DONGLE_HANDLE hDongle);

说明：本函数用于关闭已打开的加密锁

参数：

hDongle [in]: 设备句柄

返回值：

DONGLE_SUCCESS 执行成功

5.JDKey_GetDongleInfo 获取硬件信息

*DWORD WINAPI JDKey_GetDongleInfo(IN DONGLE_HANDLE hDongle, OUT DONGLE_INFO * pDI);*

说明：本函数用于获取加密锁的基本信息，调用权限：匿名

参数：

hDongle [in]: 设备句柄

pDI [out]: 硬件信息

返回值：

DONGLE_SUCCESS 执行成功

DONGLE_INFO 成员说明：

DWORD m_Ver_Hardware; //硬件版本

DWORD m_Ver_Firmware; //固件版本

DWORD m_RealTime_UTC; //UTC 实时钟 （格林威治时间，时钟锁有效）

DWORD m_SpaceSize_ISO; //ISO 空间大小

DWORD m_TickCount; //硬件上电时间

DWORD m_ProductID; //产品 ID

BYTE m_HardSN[8]; //8 字节的硬件 SN

注意：m_RealTime_UTC 项仅对带有时钟型号有效

m_ProductID 项（产品 ID）是由客户私有种子码计算生成的，具有唯一性，可用作产品的认证标识，出厂默认值为 0xFFFFFFFF

6.JDKey_VerifyPIN 校验 PIN 码

*DWORD WINAPI JDKey_VerifyPIN(IN DONGLE_HANDLE hDongle, IN int nFlag, IN char * pPIN, OUT int * pRemainCount);*

说明：本函数用于校验用户 PIN 码或管理员 PIN 码

参数：

hDongle [in]: 设备句柄

nFlag [in]: FLAG_USERPIN or FLAG_ADMINPIN

pPIN [in]: 指向 8 字节长的用户 PIN 码字串，或 16 字节长的管理员 PIN 码字串

pRemainCount [out]: 如果返回的错误码是 DONGLE_INCORRECT_PIN 时，则返回剩余的允许次数

返回值：

DONGLE_SUCCESS 执行成功

DONGLE_INCORRECT_PIN PIN 码错误

DONGLE_PIN_BLOCKED PIN 码已锁死

注意：

出厂时默认的用户 PIN 码为“12345678”

出厂时默认的管理员 PIN 码为“FFFFFFFFFFFFFFFF”

7.JDKey_ChangePIN 修改 PIN 码

*DWORD WINAPI JDKey_ChangePIN(IN DONGLE_HANDLE hDongle, IN int nFlag, IN char * pOldPIN, IN char * pNewPIN, OUT int * pRemainCount);*

说明：本函数用于更改用户 PIN 码或管理员 PIN 码

参数：

hDongle [in]: 设备句柄

nFlag [in]: FLAG_USERPIN or FLAG_ADMINPIN

pOldPIN [in]: 指向 8 字节长的旧用户 PIN 码字串，或 16 字节长的旧管理员 PIN 码字串

pNewPIN [in]: 指向 8 字节长的新用户 PIN 码字串，或 16 字节长的新管理员 PIN 码字串

pRemainCount [out]:如果返回的错误码是 DONGLE_INCORRECT_PIN 时，则返回剩余的允许次数

返回值：

DONGLE_SUCCESS	执行成功
DONGLE_INCORRECT_PIN	PIN 码错误
DONGLE_PIN_BLOCKED	PIN 码已锁死

8.JDKey_UnlockUserPIN 管理员解锁用户 PIN 码

*DWORD WINAPI JDKey_UnlockUserPIN(IN DONGLE_HANDLE hDongle, IN char * pAdminPIN, OUT int * pRemainCount);*

说明：本函数用于管理员解锁用户 PIN 码

参数：

hDongle [in]: 设备句柄

pAdminPIN [in]: 指向 16 字节长的管理员 PIN 码字串

pRemainCount [out]: 如果返回的错误码是 DONGLE_INCORRECT_PIN 时，则返回剩余的允

许次数

返回值：

DONGLE_SUCCESS 执行成功

DONGLE_INCORRECT_PIN PIN 码错误

DONGLE_PIN_BLOCKED PIN 码已锁死

9. JDKey_ResetPIN 复位 PIN 码状态到匿名

DWORD WINAPI JDKey_ResetPIN(IN DONGLE_HANDLE hDongle);

说明：本函数用于复位加密锁内的当前 PIN 码状态为匿名

参数：

hDongle [in]: 设备句柄

返回值：

DONGLE_SUCCESS 执行成功

10.JDKey_ SetPINRetries 设置 PIN 码允许重试的次数

DWORD WINAPI JDKey_SetPINRetries(IN DONGLE_HANDLE hDongle, BYTE bAdminPINRetries, BYTE bUserPINRetries);

说明：本函数用于设置 PIN 码允许重试的次数，调用权限：管理员

参数：

hDongle [in]: 设备句柄

bAdminPINRetries [in]: 管理员 PIN 码重试次数 (1-255, 其中 255 表示不限制)

bUserPINRetries [in]: 用户 PIN 码重试次数 (1-255, 其中 255 表示不限制)

返回值：

DONGLE_SUCCESS 执行成功

DONGLE_ADMINPIN_NOT_CHECK 管理员密码没有验证

注：管理员 PIN 码和用户 PIN 码允许重试次数的出厂默认值都是 255，即无限制

11.JDKey_ GenRandom 取随机数

*DWORD WINAPI JDKey_GenRandom(IN DONGLE_HANDLE hDongle, IN int len_need, OUT BYTE * pBuf);*

说明：本函数用于取硬件真随机数发生器产生的随机数，调用权限：匿名

参数：

hDongle [in]: 设备句柄

len_need [in]: 需要产生的随机数长度 (1-1024 字节)

pBuf [out]: 接收硬件产生的随机数

返回值：

DONGLE_SUCCESS 执行成功

12.JDKey_ LED LED 灯控制

DWORD WINAPI JDKey_LED(IN DONGLE_HANDLE hDongle, IN int Flag);

说明：本函数用于 LED 灯控制，调用权限：匿名

参数：

hDongle [in]: 设备句柄

Flag [in]: LED 工作模式:LED_OFF（灭）, LED_ON（亮）, LED_BLINK（闪）

返回值：

DONGLE_SUCCESS 执行成功

13.JDKey_ Read 读存储区

*DWORD WINAPI JDKey_Read(IN DONGLE_HANDLE hDongle, IN int Region, IN WORD Offset, IN WORD Length, OUT BYTE * pBuf);*

说明：本函数用于读存储区，调用权限：匿名 或 用户

参数：

hDongle [in]: 设备句柄

Region [in]: 区域（REGION_1 或 REGION_2）

Offset [in]: 偏移地址（0-0x03FF）

Length [in]: 要读出的数据长度

pBuf [in]: 接收数据的缓冲区

返回值：

DONGLE_SUCCESS 执行成功

14.JDKey_ Write 写存储区

*DWORD WINAPI JDKey_Write(IN DONGLE_HANDLE hDongle, IN int Region, IN WORD Offset, IN WORD Length, IN BYTE * pBuf);*

说明：本函数用于写存储区，调用权限：用户 或 管理员

参数：

hDongle [in]: 设备句柄

Region [in]: 区域 （REGION_1 或 REGION_2）

Offset [in]: 偏移地址 （0-0x03FF）

Length [in]: 要写入的数据长度

pBuf [in]: 要写入的数据

返回值：

DONGLE_SUCCESS 执行成功

15.JDKey_ SetKey 设置密钥

*DWORD WINAPI JDKey_SetKey(IN DONGLE_HANDLE hDongle, int HMAC_KeyID, IN BYTE * pKeyBuf, IN int Len_Key);*

说明：本函数用于设置密钥，调用权限：管理员

参数：

hDongle [in]: 设备句柄

KeyID [in]: 密钥索引:1-8

pKeyBuf [in]: 密钥

Len_Key [in]: 密钥长度 (16 字节(TEA,TDES KEY) 或 32 字节(MD5 TokenKey) 或 40 字节(SHA1 TokenKey) 或 64 字节(SHA256 TokenKey) 或 64 字节(SM3 TokenKey))

返回值：

DONGLE_SUCCESS 执行成功

DONGLE_ADMINPIN_NOT_CHECK 管理员密码没有验证

16.JDKey_ HMAC 进行 HMAC 运算

*DWORD WINAPI JDKey_HMAC(IN DONGLE_HANDLE hDongle, IN int Flag, IN int HMAC_KeyID, IN int Len_Text, IN BYTE * pText, OUT BYTE * pDigest);*

说明：本函数用于 HMAC 运算，调用权限：用户以上

参数：

hDongle [in]: 设备句柄

Flag [in]: 指示 HMAC 的 HASH 算法类型：HASH_MD5，HASH_SHA1，HASH_SHA256，HASH_SM3

KeyID [in]: 密钥索引:1-8

Len_Text [in]: 明文长度 (<=256 字节)

pText [in]: 明文数据

pDigest [out]: 输出摘要结果 (MD5 输出长度为 16 字节，SHA1 是 20 字节，SHA256 是 32 字节, SM3 也是 32 字节)

返回值：

DONGLE_SUCCESS 执行成功

DONGLE_USERPIN_NOT_CHECK 用户密码没有验证

17. JDKey_TEA 进行 TEA 加解密运算

*DWORD WINAPI JDKey_TEA(IN DONGLE_HANDLE hDongle, IN int KeyID, IN int Flag, IN BYTE * pInData, OUT BYTE * pOutData, IN int DataLen);*

说明：本函数用于 TEA 加解密运算，调用权限：用户以上

参数：

hDongle [in]: 设备句柄

KeyID [in]: 密钥索引:1-8 (使用密钥中的前 16 字节作为运算密钥)

Flag [in]: 运算方式: FLAG_ENCODE 表示加密运算,FLAG_DECODE 表示解密运算

pInData [in]: 输入数据

pOutData [out]: 输出数据缓冲区，大小至少要和输入数据缓冲区相同，输入和输出数据缓冲区可以为同一个

DataLen [in]: 输入数据的长度，必须是 8 的整数倍

返回值：

DONGLE_SUCCESS 执行成功

DONGLE_USERPIN_NOT_CHECK 用户密码没有验证

注意：本函数运算模式是 ECB 模式

18. JDKey_TDES 进行 TDES 加解密运算

*DWORD WINAPI JDKey_TDES(IN DONGLE_HANDLE hDongle, IN int KeyID, IN int Flag, IN BYTE * pInData, OUT BYTE * pOutData, IN int DataLen);*

说明：本函数用于 TDES 加解密运算，调用权限：用户以上

参数：

hDongle [in]: 设备句柄

KeyID [in]: 密钥索引:1-8 (使用密钥中的前 16 字节作为运算密钥)

Flag [in]: 运算方式: FLAG_ENCODE 表示加密运算,FLAG_DECODE 表示解密运算

pInData [in]: 输入数据

pOutData [out]: 输出数据缓冲区，大小至少要和输入数据缓冲区相同，输入和输出数据

缓冲区可以为同一个

DataLen [in]: 输入数据的长度，必须是 8 的整数倍

返回值：

DONGLE_SUCCESS 执行成功

DONGLE_USERPIN_NOT_CHECK 用户密码没有验证

注意：本函数运算模式是 ECB 模式

19.JDKey_ Setup 设置加密锁

*DWORD WINAPI JDKey_Setup(IN DONGLE_HANDLE hDongle, IN BYTE * pSeed, IN int Len_Seed, OUT char * pProductID, OUT char * pAdminPIN);*

说明：本函数用于设置加密锁，调用权限:管理员

参数：

hDongle [in]: 设备句柄

pSeed [in]: 指向不超过 256 字节的种子码

Len_Seed [in]: 种子码长度

pProductID [out]: 输出 8 字节的产品 ID 字串

pAdminPIN [out]: 输出 16 字节的管理员 PIN 码字串

返回值：

DONGLE_SUCCESS 执行成功

DONGLE_ADMINPIN_NOT_CHECK 管理员密码没有验证

说明：由种子码产生出产品 ID 和管理员 PIN 码的过程是单向的和唯一性的，请注意对种子码的妥善保管和保密

20.JDKey_RFS 恢复出厂设置

DWORD WINAPI JDKey_RFS(IN DONGLE_HANDLE hDongle);

说明：本函数用于把加密锁恢复到出厂时的状态，调用权限：管理员

参数：

hDongle [in]: 设备句柄

返回值：

DONGLE_SUCCESS 执行成功

DONGLE_ADMINPIN_NOT_CHECK 管理员密码没有验证

注意：本函数调用后硬件会自动复位重启

21. JDKEY_SOFT_HMAC 服务器端 HMAC 计算

*BOOL WINAPI JDKEY_SOFT_HMAC(IN int flag, IN BYTE * pucText, IN int ulText_Len, IN BYTE * pucKey, IN int ulKey_Len, OUT BYTE * pucTokenKey, OUT BYTE * pucDigest);*

说明：本函数用于服务器端 HMAC 计算

参数：

Flag [in]: 指示 HMAC 运算的 HASH 算法类型：HASH_MD5，HASH_SHA1，HASH_SHA256，HASH_SM3

pucText [in]: 指向明文数据流

ulText_Len [in]: 明文数据流长度

pucKey [in]: 身份认证密钥

ulKey_Len [in]: 身份认证密钥的长度

pucTokenKey [out]: 输出锁中密钥(对 MD5 长度是 32 字节, SHA1 是 40 字节, SHA256 是 64 字节, SM3 是 64 字节)

pucDigest [out]: 输出明文的 hmac 值(md5 是 16 字节, sha1 是 20 字节, sha256 是 32 字节, sm3 是 32 字节)

返回值：

DONGLE_SUCCESS 执行成功

22. JDKey_SOFT_TEA 软件 TEA 运算

*BOOL WINAPI JDKey_SOFT_TEA(IN BYTE * pucKey, IN int Flag, IN BYTE * pInData, OUT BYTE * pOutData, IN int DataLen);*

说明：本函数用于软件 TEA 运算

参数：

pucKey [in]: 指向 16 字节长的密钥

Flag [in]: 运算方式: FLAG_ENCODE 表示加密运算,FLAG_DECODE 表示解密运算

pInData [in]: 输入数据

pOutData [out]: 输出数据缓冲区，大小至少要和输入数据缓冲区相同，输入和输出数据缓冲区可以为同一个

DataLen [in]: 输入数据的长度，必须是 8 的整数倍

返回值：

DONGLE_SUCCESS 执行成功

注意：本函数运算模式是 ECB 模式

23. JDKey_SOFT_TDES 软件 TDES 运算

*BOOL WINAPI JDKey_SOFT_TDES(IN BYTE * pucKey, IN int Flag, IN BYTE * pInData, OUT BYTE * pOutData, IN int DataLen);*

说明：本函数用于软件 TDES 运算

参数：

pucKey [in]: 指向 16 字节长的密钥

Flag [in]: 运算方式: FLAG_ENCODE 表示加密运算,FLAG_DECODE 表示解密运算

pInData [in]: 输入数据

pOutData [out]: 输出数据缓冲区，大小至少要和输入数据缓冲区相同，输入和输出数据缓冲区可以为同一个

DataLen [in]: 输入数据的长度，必须是 8 的整数倍

返回值：

DONGLE_SUCCESS 执行成功

注意：本函数运算模式是 ECB 模式

24. JDKey_ISO_Download 下载 ISO

DWORD WINAPI JDKey_ISO_Download(IN DONGLE_HANDLE hDongle, IN BYTE pISO, IN DWORD len_ISO);*

说明：本函数用于下载 ISO,仅适用于光盘锁 调用权限：管理员

参数：

hDongle [in]: 设备句柄

pISO [in]: 指向 ISO 文件数据

len_ISO [in]: 指向 ISO 文件长度，必须小于 128K

返回值：

DONGLE_SUCCESS 执行成功

DONGLE_ADMINPIN_NOT_CHECK 管理员密码没有验证

三、常量与错误码定义

```
//加密锁句柄定义
typedef void *    DONGLE_HANDLE;

//默认密码
#define CONST_USERPIN        "12345678"              //出厂时默认的 USERPIN
#define CONST_ADMINPIN       "FFFFFFFFFFFFFFFF" //出厂时默认的 ADMINPIN

//LED 灯控制
#define LED_OFF              0 //灯灭
#define LED_ON               1 //灯亮
#define LED_BLINK            2 //1hz 闪

//PIN 码类型
#define FLAG_USERPIN         0 //校验用户 PIN
#define FLAG_ADMINPIN        1 //校验开发商 PIN

//HASH 算法类型
#define HASH_MD5             0 //MD5
#define HASH_SHA1            1 //SHA1
#define HASH_SHA256          2 //SHA256
#define HASH_SM3             3 //SM3

//区域标识
#define REGION_1         0 //用    户存储区,大小:1024 字节,权限:匿名可读,用户以上可写
#define REGION_2         1 //管理员存储区,大小:1024 字节,权限:用户可读,    管理员可写
```

```
//运算方式
#define FLAG_ENCODE        0 //加密运算
#define FLAG_DECODE        1 //解密运算

//错误码
#define DONGLE_SUCCESS                0x00000000        // 操作成功
#define DONGLE_NOT_FOUND              0x80000001        // 未找到指定的加密锁
#define DONGLE_INVALID_HANDLE         0x80000002        // 无效的句柄
#define DONGLE_INVALID_PARAMETER      0x80000003        // 参数错误
#define DONGLE_FAILED                 0x80000004        // 操作失败
#define DONGLE_NOT_SUPPORT            0x80000005        // 操作不支持
#define DONGLE_SESSIONKEY_FAILED      0x80000007        // 协商通信密钥失败
#define DONGLE_SESSIONKEY_NOT_GEN     0x80000008        // 尚未协商通信密钥
#define DONGLE_ADMINPIN_NOT_CHECK     0x8000000A        // 管理员密码没有验证
#define DONGLE_USERPIN_NOT_CHECK      0x8000000B        // 用户密码没有验证
#define DONGLE_PIN_BLOCKED            0x8000000C        // PIN 码已锁死
#define DONGLE_INCORRECT_PIN          0x8000000D        // PIN 码错误
#define DONGLE_COMM_ERROR             0x80000012        // 通讯错误
#define DONGLE_ERROR_UNKNOWN          0xFFFFFFFF        // 未知的错误

//加密锁硬件信息结构
typedef struct{
    DWORD       m_Ver_Hardware;         //硬件版本
    DWORD       m_Ver_Firmware;         //固件版本
    DWORD       m_RealTime_UTC;         //UTC 实时钟
    DWORD       m_SpaceSize_ISO;        //ISO 空间大小
    DWORD       m_TickCount;            //硬件上电时间
    DWORD       m_ProductID;            //产品 ID
    BYTE        m_HardSN[8];            //8 字节的硬件 SN
} DONGLE_INFO;
```