

# ET99 扩展 API 接口

通过 ET99 的扩展 API 接口可以完成设置许可证，验证许可证，3DES 加解密的操作，为广大软件加密厂商提供了更多，更强大的加密手段。

## 1、设置许可证：

ET99 可以设置 7 个许可证。许可证为 16 个任意二进制字节，可以是不可显示字符。设置许可证需要输入正确的 PID 和 UserPIN。

```
long _stdcall ET99App_SetLicense
(
    char* pcPID,      //ET99的PID（8个字节）
    char* pcUserPIN,  //ET99的UserPIN（16个字节）
    int iIndex,       //许可证号（1—7）
    char* pcLicense   //许可证内容（16个字节）
);
```

**说明：**

设置许可证。

**参数：**

pcPID            [in]ET99 的 PID，为 8 个十六进制字符，如“FFC5EB78”。

pcUserPIN       [in] ET99 的 UserPIN，为 16 个十六进制字符，如“FFFFFFFFFFFFFFFF”。

iIndex           [in]许可证编号，范围从 1—7。

pcLicense       [in]许可证内容，为 16 个任意二进制字节，可以是不可显示字符。

**返回值：**

返回0x00000000表示成功。返回其它值表示执行该函数失败，参看文档第6部分错误编码。

## 2、检查许可证是否有效：

检查设置的许可证是否有效。许可证为 16 个任意二进制字节，可以是不可显示字符。检查

许可证是否有效时需要输入正确的 PID 和 UserPIN。

```
long _stdcall ET99App_CheckLicense
(
    char* pcPID,      //ET99的PID (8个字节)
    char* pcUserPIN, //ET99的UserPIN (16个字节)
    int iIndex,       //许可证号 (1—7)
    char* pcLicense //许可证内容 (16个字节)
);
```

**说明：**

检查许可证是否有效。

**参数：**

pcPID [in]ET99 的 PID，为 8 个十六进制字符，如“FFC5EB78”。

pcUserPIN [in] ET99 的 UserPIN，为 16 个十六进制字符，如“FFFFFFFFFFFFFFFF”。

iIndex [in]许可证编号，范围从 1—7。

pcLicense [in]许可证内容，为 16 个任意二进制字节，可以是不可显示字符。

**返回值：**

返回0x00000000表示成功。返回其它值表示执行该函数失败，参看文档第6部分错误编码。

### 3、设置 3DES 密钥：

设置 3DES 加解密密钥。3DES 密钥为长度 24 字节（192 位）。设置 3DES 密钥时需要输入正确的 PID 和 UserPIN。

```
long _stdcall ET99App_Set3DESKey
(
    char* pcPID,      //ET99的PID (8个字节)
    char* pcUserPIN, //ET99的UserPIN (16个字节)
    char* pc3DESKey //3DES密钥 (24个字节)
);
```

**说明：**

设置 3DES 密钥。

**参数：**

pcPID [in]ET99 的 PID，为 8 个十六进制字符，如“FFC5EB78”。

pcUserPIN [in] ET99 的 UserPIN，为 16 个十六进制字符，如“FFFFFFFFFFFFFFFF”。

pc3DESKey [in]3DES 密钥，为 24 个任意二进制字节，可以是不可显示字符。

**返回值：**  
返回0x00000000表示成功。返回其它值表示执行该函数失败，参看文档第6部分错误编码。

**4、3DES 加密：**

使用 3DES 算法（CBC 模式）对数据进行加密。3DES 密钥为长度 24 字节（192 位）。3DES 加密时需要输入正确的 PID 和 UserPIN。

```
long _stdcall ET99App_3DESEnc
(
    char* pcPID,      //ET99的PID（8个字节）
    char* pcUserPIN,  //ET99的UserPIN（16个字节）
    char* pcIV,       //3DES加密的初始向量iv值（8个字节）
    char* pcData,     //待加密的数据缓冲区
    int iDataLen      //待加密的数据长度（必须是8的倍数）
);
```

**说明：**  
3DES 加密。

**参数：**  
pcPID            [in]ET99 的 PID，为 8 个十六进制字符，如“FFC5EB78”。  
pcUserPIN       [in] ET99 的 UserPIN，为 16 个十六进制字符，如“FFFFFFFFFFFFFFFF”。  
PcIV            [in] 3DES 加密的初始向量 iv 值。iv 不同，加密的结果就不同。  
pcData           [in/out] 双向缓冲区，输入为待加密的数据，输出为加密结果。  
IDataLen        [in] 待加密的数据长度，必须是 8 的倍数。

**返回值：**  
返回0x00000000表示成功。返回其它值表示执行该函数失败，参看文档第6部分错误编码。

**5、3DES 解密：**

使用 3DES 算法（CBC 模式）对数据进行解密。3DES 密钥为长度 24 字节（192 位）。3DES 解密时需要输入正确的 PID 和 UserPIN。

```
long _stdcall ET99App_3DESDec
(
    char* pcPID,      //ET99的PID（8个字节）
    char* pcUserPIN, //ET99的UserPIN（16个字节）
    char* pcIV,       //3DES解密的初始向量iv值（8个字节）
    char* pcData,     //待解密的数据缓冲区
    int iDataLen      //待解密的数据长度（必须是8的倍数）
);
```

**说明：**  
3DES 解密。

**参数：**

- pcPID [in]ET99 的 PID，为 8 个十六进制字符，如“FFC5EB78”。
- pcUserPIN [in] ET99 的 UserPIN，为 16 个十六进制字符，如“FFFFFFFFFFFFFFFF”。
- PcIV [in] 3DES 解密的初始向量 iv 值。iv 不同，解密的结果就不同。
- pcData [in/out] 双向缓冲区，输入为待解密的数据，输出为解密结果。
- IDataLen [in] 待解密的数据长度，必须是 8 的倍数。

**返回值：**  
返回0x00000000表示成功。返回其它值表示执行该函数失败，参看文档第6部分错误编码。

6、错误编码

值	说明
0x00	函数执行成功
0x01	访问被拒绝，权限不够
0x02	通讯错误，没有打开设备
0x03	无效的参数，参数出错
0x04	没有设置 PID
0x05	打开指定的设备失败
0x06	硬件错误
0x07	未知错误
0x08	输入错误
0x09	没有找到 ET99
0x10	许可证无效

## 7、调用示例

```
#include <stdio.h>
#include "ET99License.h"

#pragma comment(lib,"ET99License.lib")

int main(int argc, char* argv[])
{
    //ET99 的 PID
    char pcPID[9] = "FFC5EB78";

    //ET99 的 UserPIN
    char pcUserPIN[17] = "FFFFFFFFFFFFFFFFF";

    //许可证为 16 个任意字符
    char pcLicense[17] = "ET99LiceseTest99";
    long lRet = 0;

    //设置许可证，许可证为 16 个任意二进制，可以是不可显示字符
    lRet = ET99App_SetLicense(pcPID, pcUserPIN, 1, pcLicense);
    if(lRet != 0)
    {
        printf("ET99App_SetLicense Error:0x%08X\n",lRet);
        return -1;
    }else
        printf("ET99App_SetLicense Sucess\n");

    //检查许可证是否有效
    lRet = ET99App_CheckLicense(pcPID, pcUserPIN, 1, pcLicense);
    if(lRet != 0)
    {
        printf("ET99App_CheckLicense Error:0x%08X\n",lRet);
        return -1;
    }else
        printf("ET99App_CheckLicense Sucess\n");

    //设置 3DES 密钥，3DES 密钥为 24 个任意二进制，可以是不可显示字符
    char pc3DESKey[25] = "123456781234567812345678";
    lRet = ET99App_Set3DESKey(pcPID, pcUserPIN, pc3DESKey);
    if(lRet != 0)
    {
        printf("ET99App_Set3DESKey Error:0x%08X\n",lRet);
```

```

        return -1;
    }else
        printf("ET99App_Set3DESKey Sucess\n");

char pbData[16] = {0};
char buf_iv[8] = {0x1,0x2,0x3,0x4,0x5,0x6,0x7,0x8};

int i =0;

for(i=0; i<16; ++i)
    pbData[i] = i;

//3DES 加密
//加密数据长度要是 8 的倍数
//pbData 是双向缓冲区，输入为待加密的数据，输出为加密结果
lRet = ET99App_3DESEnc(pcPID, pcUserPIN, buf_iv, pbData, 16);
if(lRet != 0)
{
    printf("ET99App_3DESEnc Error:0x%08X\n",lRet);
    return -1;
}else
    printf("ET99App_3DESEnc Sucess. Cipher Data:\n");

for(i=0; i<16; ++i)
{
    printf("%02X ", (unsigned char)pbData[i]);
}

//3DES 解密
//解密数据长度要是 8 的倍数
//pbData 是双向缓冲区，输入为待解密的数据，输出为解密结果
lRet = ET99App_3DESDec(pcPID, pcUserPIN, buf_iv, pbData, 16);
if(lRet != 0)
{
    printf("\nET99App_3DESEnc Error:0x%08X\n",lRet);
    return -1;
}else
    printf("\nET99App_3DESEnc Sucess. Plain Data:\n");

for(i=0; i<16; ++i)
{
    printf("%02X ", (unsigned char)pbData[i]);
}

```

```
printf("\n");  
return 0;  
}
```