GccaCryptEx

Функция **GccaCryptEx** зашифровывает или расшифровывает блок данных при помощи аппаратного или программно-реализованного алгоритма.

C

```
int GccaCryptEx(
  HANDLE hGrd,
  DWORD dwAlgo,
  DWORD dwDataLng,
  void *pData,
  DWORD dwMethod,
  DWORD dwIVLng,
  void *pIV,
  void *pKeyBuf,
  void *pContext,
  void *pReserved
);
```

hGrd	не используется
dwAlgo	1) номер аппаратного алгоритма **AES128,** если алгоритм вызывается через дескриптор. Или: 2) если ключ шифрования задается напрямую, а не считывается из дескриптора, номер алгоритма должен быть **GrdSC_AES128**.
dwDataLng	длина блока данных в байтах
pData	буфер данных для преобразования
dwMethod	метод преобразования, который задается комбинацией флагов **GrdAM_XXX** и **GrdSC_XXX** <table><tr><td colspan="2">**Биты 0-5 - режим работы алгоритма**</td></tr><tr><td>GrdAM_ECB</td><td>Режим электронной кодовой книги</td></tr><tr><td>GrdAM_CBC</td><td>Режим сцепления кодированных блоков</td></tr><tr><td>GrdAM_CFB</td><td>Режим с кодированной обратной связью</td></tr><tr><td>GrdAM_OFB</td><td>Режим с обратной связью по выходу</td></tr><tr><td colspan="2">**Бит 6 - резерв**</td></tr><tr><td colspan="2">**Бит 7 - тип операции**</td></tr><tr><td>GrdAM_Encrypt</td><td>Кодировать блок</td></tr><tr><td>GrdAM_Decrypt</td><td>Декодировать блок</td></tr><tr><td colspan="2">**Биты 8-9: тип блока данных**</td></tr><tr><td>GrdSC_First</td><td>Первый блок данных</td></tr><tr><td>GrdSC_Next</td><td>Следующий блок данных</td></tr><tr><td>GrdSC_Last</td><td>Последний блок данных</td></tr><tr><td>GrdSC_All</td><td>Единственный блок данных</td></tr></table>
dwIVLng	длина вектора инициализации: 16 байтов
pIV	вектор инициализации
pKeyBuf	буфер для передачи ключа шифрования для алгоритма (AES). Длина ключа 128 бит (16 байт).

<table>
<tr><td>pContext</td><td>буфер для контекста при шифровании больших массивов данных, которые разбиваются на несколько блоков. Для контекста должна быть зарезервирована память размером GrdXXXXXX_CONTEXT_SIZE байт в зависимости от алгоритма. Только для программно-реализованных алгоритмов. При использовании аппаратного алгоритма параметр должен быть равен NULL
<table>
<tr><td>GrdAES128_KEY_SIZE</td><td>Длина ключа AES - 128 бит</td></tr>
<tr><td>GrdAES128_BLOCK_SIZE</td><td>Длина блока данных AES - 128 бит</td></tr>
<tr><td>GrdAES_CONTEXT_SIZE</td><td>Значение должно быть больше или равно sizeof(AES_CONTEXT)</td></tr>
</table></td></tr>
<tr><td>Reserved</td><td>не используется. Параметр должен быть равен NULL</td></tr>
</table>

Набор ошибок Guardant API

Функция **GccaCryptEx** позволяет зашифровывать и расшифровывать данные с помощью алгоритма **AES128. GccaCryptEx** предназначена для работы с алгоритмами шифрования с переменным вектором инициализации.

Преобразование производится алгоритмом с порядковым номером, заданным в параметре *dwAlgo*. В зависимости от номера алгоритма *dwAlgo* ф нкция определяет, каким образом реализован алгоритм - аппаратным или программным.

Программно-реализованные алгоритмы шифрования при шифровании больших массивов данных используют контекст, память для которого размером не менее **GrdXXXXXX_CONTEXT_SIZE** должна быть зарезервирована до вызова функции. Указатель на буфер для контекста передается через параметр *pContext*.

Длина шифруемых блоков данных зависит от метода шифрования (см. описание методов в *Руководство пользователя, часть 3*). Для методов **CFB** и **OFB** длина шифруемых блоков может быть произвольной.

Если в дескрипторе аппаратного алгоритма установлен флаг **nsafl_GP_dec** (уменьшение счетчика), вычитание счетчика GP происходит при каждом вызове **GccaCryptEx**.

Новый параметр (*dwIVLng*) имеет смысл для аппаратных алгоритмов с переменным вектором инициализации, которые появятся в будущем. На существующих алгоритмах это отразится лишь в том случае, если указывается длина вектора инициализации от 0 до 8 байт (включительно). При указании длины более 8 байт шифрование происходит с использованием первых 8 байт указанного вектора инициализации.

При вызове **GccaCryptEx** с нулевым указателем на вектор инициализации возвращается GrdE_OK, шифрование и последующее расшифрование происходит нормально в любых режимах (в т. ч. тех, которые требуют вектор инициализации). Ситуация полностью аналогична использованию нулевого вектора инициализации или вектора инициализации нулевой длины. Может работать как через защищенные ячейки типа алгоритм, так и напрямую обращаться к алгоритму AES128 (GrdSC_AES128 в параметре *dwAlgo*).