

GrdTransform

Функция(метод) **GrdTransform** преобразует блок данных при помощи аппаратного алгоритма.

C

```
int GRD_API GrdTransform(
    HANDLE hGrd,
    DWORD dwAlgoNum,
    DWORD dwLng,
    void *pData,
    DWORD dwMethod,
    void *pIV
);
```

<i>hGrd</i>	хэндл, через который будет выполнена данная операция																		
<i>dwAlgoNum</i>	номер аппаратного алгоритма с помощью которого будет производиться шифрование или расшифрование																		
<i>dwLng</i>	длина буфера данных в байтах																		
<i>pData</i>	указатель на буфер данных открытого или зашифрованного текста																		
<i>dwMethod</i>	следующие флаги позволяют указать режим работы алгоритма и тип операции: <table><tr><td colspan="2">Биты 0-5 - режим работы алгоритма</td></tr><tr><td>GrdA_M_ECB</td><td>Режим электронной кодовой книги (режим простой замены). Каждый блок открытого текста заменяется блоком шифротекста. Шифрование двух одинаковых блоков даст идентичный результат. Скорость обработки блоков в режиме ECB фиксирована. Недостаток ECB, в сравнении с другими режимами шифрования, — сохранение статистических особенностей открытого текста.</td></tr><tr><td>GrdA_M_CBC</td><td>Режим сцепления блоков шифротекста. Каждый блок открытого текста (кроме первого) побитово складывается по модулю 2 (операция XOR) с предыдущим результатом шифрования. Таким образом, каждый блок зашифрованного текста зависит от всех блоков открытого текста, обработанных до него. Режим CBC лишён недостатка алгоритма ECB, но всё же имеет ряд недостатков с точки зрения безопасности.</td></tr><tr><td>GrdA_M_CFB</td><td>Режим обратной связи по шифротексту (режим гаммирования с обратной связью). Для шифрования следующего блока открытого текста он складывается по модулю 2 с перешифрованным (блочным шифром) результатом шифрования предыдущего блока. Криптостойкость CFB определяется криптостойкостью используемого шифра.</td></tr><tr><td>GrdA_M_OFB</td><td>Режим обратной связи по выходу. В этом режиме открытый текст используются только для конечного сложения. Операции блочного шифра могут быть выполнены заранее, позволяя выполнить заключительное шифрование параллельно с открытым текстом.</td></tr><tr><td colspan="2">Бит 6 - резерв</td></tr><tr><td colspan="2">Бит 7 - тип операции</td></tr><tr><td>GrdA_M_Encode</td><td>Зашифровать блок данных</td></tr><tr><td>GrdA_M_Decode</td><td>Расшифровать блок данных</td></tr></table> Флаги GrdSC_XXX тут не используются, так как это просто синоним старой функции nXkTransformEx. Мы рекомендуем вместо нее использовать функцию GrdCrypt.	Биты 0-5 - режим работы алгоритма		GrdA_M_ECB	Режим электронной кодовой книги (режим простой замены). Каждый блок открытого текста заменяется блоком шифротекста. Шифрование двух одинаковых блоков даст идентичный результат. Скорость обработки блоков в режиме ECB фиксирована. Недостаток ECB, в сравнении с другими режимами шифрования, — сохранение статистических особенностей открытого текста.	GrdA_M_CBC	Режим сцепления блоков шифротекста. Каждый блок открытого текста (кроме первого) побитово складывается по модулю 2 (операция XOR) с предыдущим результатом шифрования. Таким образом, каждый блок зашифрованного текста зависит от всех блоков открытого текста, обработанных до него. Режим CBC лишён недостатка алгоритма ECB, но всё же имеет ряд недостатков с точки зрения безопасности.	GrdA_M_CFB	Режим обратной связи по шифротексту (режим гаммирования с обратной связью). Для шифрования следующего блока открытого текста он складывается по модулю 2 с перешифрованным (блочным шифром) результатом шифрования предыдущего блока. Криптостойкость CFB определяется криптостойкостью используемого шифра.	GrdA_M_OFB	Режим обратной связи по выходу. В этом режиме открытый текст используются только для конечного сложения. Операции блочного шифра могут быть выполнены заранее, позволяя выполнить заключительное шифрование параллельно с открытым текстом.	Бит 6 - резерв		Бит 7 - тип операции		GrdA_M_Encode	Зашифровать блок данных	GrdA_M_Decode	Расшифровать блок данных
Биты 0-5 - режим работы алгоритма																			
GrdA_M_ECB	Режим электронной кодовой книги (режим простой замены). Каждый блок открытого текста заменяется блоком шифротекста. Шифрование двух одинаковых блоков даст идентичный результат. Скорость обработки блоков в режиме ECB фиксирована. Недостаток ECB, в сравнении с другими режимами шифрования, — сохранение статистических особенностей открытого текста.																		
GrdA_M_CBC	Режим сцепления блоков шифротекста. Каждый блок открытого текста (кроме первого) побитово складывается по модулю 2 (операция XOR) с предыдущим результатом шифрования. Таким образом, каждый блок зашифрованного текста зависит от всех блоков открытого текста, обработанных до него. Режим CBC лишён недостатка алгоритма ECB, но всё же имеет ряд недостатков с точки зрения безопасности.																		
GrdA_M_CFB	Режим обратной связи по шифротексту (режим гаммирования с обратной связью). Для шифрования следующего блока открытого текста он складывается по модулю 2 с перешифрованным (блочным шифром) результатом шифрования предыдущего блока. Криптостойкость CFB определяется криптостойкостью используемого шифра.																		
GrdA_M_OFB	Режим обратной связи по выходу. В этом режиме открытый текст используются только для конечного сложения. Операции блочного шифра могут быть выполнены заранее, позволяя выполнить заключительное шифрование параллельно с открытым текстом.																		
Бит 6 - резерв																			
Бит 7 - тип операции																			
GrdA_M_Encode	Зашифровать блок данных																		
GrdA_M_Decode	Расшифровать блок данных																		
<i>pIV</i>	8-байтовый вектор инициализации. Если в качестве указателя на вектор инициализации задан NULL, то преобразование пройдет корректно, при этом будет использован нулевой вектор Для алгоритмов Guardant Stealth I и Fidus параметр не используются (значение должно быть NULL)																		
GrdE_AlgoNotFound	Алгоритм с указанным номером не существует																		
GrdE_CRCErrorFunc	Ошибка CRC при выполнении алгоритма. Эта ошибка обычно возникает, если длина преобразуемой последовательности не совпадает с заданной во время создания алгоритма																		
GrdE_GPIs0	Счетчик алгоритма достиг нулевого значения. Результат алгоритма больше нельзя получить																		

Функции **GrdTransform** позволяют преобразовывать информацию аппаратным алгоритмом ключа. Преобразование производится алгоритмом с порядковым номером, заданным в параметре *dwAlgoNum*. Этот алгоритм предварительно должен быть создан. Если в дескрипторе алгоритма установлен флаг **nsafl_GP_dec** (уменьшение счетчика), вычитание счетчика GP происходит при каждом вызове **GrdTransform**.

Если шифрование выполнялось блоками произвольной длины, то для корректного расшифрования длина и порядок обработки блоков должны сохраняться.

Для алгоритмов GSII64

Длина массива преобразуемых данных (в байтах) *pData* задается параметром *dwLng* и зависит от режима работы алгоритма. Для режимов ECB и CBC длина данных должна быть кратной **GrdARS_GSII64** (8 байт), максимально **GrdAMRS_GSII64** (248 байт). Если задана длина массива, не кратная 8 байтам, функция возвращает код ошибки [GrdE_InvalidArg](#). Для режимов CFB и OFB длина может быть произвольной, но не превышающей 255 байт. Параметр *dwMethod* задается суммой флагов (см. GRDAPL.H).

Массив данных для преобразования должен находиться по адресу, указанному в параметре *pData*. Если функция выполнена успешно, по этому же адресу будет помещена последовательность преобразованных данных той же длины. В этом случае функция возвращает [GrdE_OK](#).

Скорость кодирования/декодирования напрямую зависит от длины *dwLng* блока данных *pData*, передаваемого в **GrdTransform**. Максимальная скорость достигается при максимальной длине блока. Если размер блока данных сильно превышает максимальное значение *dwLng*, его нужно разбивать на куски максимальной возможной длины. Однако при таком подходе ключ (особенно с интерфейсом LPT) может оказываться занятым на более долгое время в течении каждой такой операции. Поэтому в приложениях, для которых это критично (например, со множественными независимыми параллельными запросами к ключу), лучше использовать более мелкие блоки.

Для режимов, использующих сцепление блоков, необходимо задавать 8-байтовый вектор инициализации *pIV*. При выполнении кодирования и декодирования необходимо задавать один и тот же вектор инициализации. Если кодирование/декодирование происходит в несколько приемов, то в качестве вектора инициализации нужно задавать значение, которое возвращается в *pIV* после выполнения предыдущей операции.

Для алгоритмов AES128

Для работы с аппаратными алгоритмами AES128 используйте функцию [GrdTransformEx](#), т. к. только в этом случае есть возможность задания полного вектора инициализации (16 байт).

Для алгоритмов HASH64

Для работы с аппаратными алгоритмами HASH64 используйте функцию [GrdHash](#).

Для алгоритмов Guardant Stealth I и Guardant Fidus

Предупреждение

Данный тип алгоритмов не отвечает современным требованиям безопасности. Поддержка этих алгоритмов реализована исключительно в целях совместимости. В связи с этим алгоритмы данного типа не рекомендуются для использования в приложениях. Следует пользоваться алгоритмами GSII64, реализованными в Guardant Stealth/Net II/III.

Преобразование однонаправленное, т. е. для функции $F(X)$ не существует функция $F^{-1}()$ такая, что $X = F^{-1}(F(X))$. Оно производится алгоритмом с порядковым номером, заданным в параметре *dwAlgoNum*. Этот алгоритм предварительно должен быть создан. Длина массива с преобразуемыми данными (в байтах) задается параметром *dwLng*. Она должна быть равной длине вопроса алгоритма, заданной в его дескрипторе (допустимые значения - от 4 до 248). Сам массив должен находиться по адресу, указанному в параметре *pData*. Если функция выполнена успешно, по этому же адресу будет помещена последовательность преобразованных данных той же длины. В этом случае функция возвращает [GrdE_OK](#).

C#

```
public static GrdE GrdTransform(Handle grdHandle, GrdAlgoNum algNum, byte[] data, GrdAM method, ref long iv)
public static GrdE GrdTransform(Handle grdHandle, GrdAlgoNum algNum, byte[] data, GrdAM method, byte[] iv)
```

grdHandle [in]

Тип: [Handle](#)

хэндл, через который будет выполнена данная операция.

algNum [in]

Тип: [GrdAlgoNum](#)

Номер аппаратного алгоритма, с помощью которого будет происходить преобразование.

data [in]

Тип: byte []

Буфер данных для преобразования.

method [in]

Тип: [GrdAM](#)

Метод преобразования. Задается комбинацией флагов [GrdAM](#).

iv [in]

Типы: byte [], long []

Вектор инициализации.

GrdE.AlgoNotFound	Алгоритм с указанным номером не существует
GrdE.CRCErrFunc	Ошибка CRC при выполнении алгоритма. Эта ошибка обычно возникает, если длина преобразуемой последовательности не совпадает с заданной во время создания алгоритма
GrdE.GPis0	Счетчик алгоритма достиг нулевого значения. Результат алгоритма больше нельзя получить

Метод **GrdTransform** позволяет преобразовывать информацию аппаратным алгоритмом ключа. Преобразование производится алгоритмом с порядковым номером, заданным в параметре *algNum*. Этот алгоритм предварительно должен быть создан. Если в дескрипторе алгоритма установлен флаг **nsafi_GP_dec** (уменьшение счетчика), вычитание счетчика GP происходит при каждом вызове **GrdTransform**.

Если шифрование выполнялось блоками произвольной длины, то для корректного расшифрования длина и порядок обработки блоков должны сохраняться.

Для алгоритмов GSII64

Длина массива преобразуемых данных (в байтах) *data* зависит от режима работы алгоритма. Для режимов ECB и CBC длина данных должна быть кратной **GrdARS_GSII64** (8 байт), максимально **GrdAMRS_GSII64** (248 байт). Для режимов CFB и OFB длина может быть произвольной, но не превышающей 255 байт. Параметр *method* задается комбинацией флагов [GrdAM](#).

Массив данных для преобразования должен находиться по адресу, указанному в параметре *data*. Если метод выполнен успешно, по этому же адресу будет помещена последовательность преобразованных данных той же длины. В этом случае метод возвращает [GrdE.OK](#).

Скорость кодирования/декодирования напрямую зависит от длины блока данных *data*, передаваемого в **GrdTransform**. Максимальная скорость достигается при максимальной длине блока. Если размер блока данных сильно превышает максимальное значение, его нужно разбивать на куски максимально возможной длины. Однако при таком подходе ключ (особенно с интерфейсом LPT) может оказываться занятым на более долгое время в течении каждой такой операции. Поэтому в приложениях, для которых это критично (например, со множественными независимыми параллельными запросами к ключу), лучше использовать более мелкие блоки.

Для режимов, использующих сцепление блоков, необходимо задавать 8-байтовый вектор инициализации *iv*. При выполнении кодирования и декодирования необходимо задавать один и тот же вектор инициализации. Если кодирование/декодирование происходит в несколько приемов, то в качестве вектора инициализации нужно задавать значение, которое возвращается в *iv* после выполнения предыдущей операции.

Для алгоритмов AES128

Для работы с аппаратными алгоритмами AES128 используйте функцию [GrdTransformEx](#), т. к. только в этом случае есть возможность задания полного вектора инициализации (16 байт).

Для алгоритмов HASH64

Для работы с аппаратными алгоритмами HASH64 используйте функцию [GrdHash](#).

Для алгоритмов Guardant Stealth I и Guardant Fidus

Предупреждение

Данный тип алгоритмов не отвечает современным требованиям безопасности. Поддержка этих алгоритмов реализована исключительно в целях совместимости. В связи с этим алгоритмы данного типа не рекомендуются для использования в приложениях. Следует пользоваться алгоритмами GSII64, реализованными в Guardant Stealth/Net II/III.

Преобразование однонаправленное, т. е. для функции $F(X)$ не существует функция $F^{-1}()$ такая, что $X=F^{-1}(F(X))$. Оно производится алгоритмом с порядковым номером, заданным в параметре *algNum*. Этот алгоритм предварительно должен быть создан. Длина массива с преобразуемыми данными (в байтах) должна быть равной длине вопроса алгоритма, заданной в его дескрипторе (допустимые значения - от 4 до 248). Сам массив должен находиться по адресу, указанному в параметре *data*. Если метод выполнен успешно, по этому же адресу будет помещена последовательность преобразованных данных той же длины. В этом случае метод возвращает [GrdE.OK](#)

Java

```
public static GrdE GrdTransform(Handle grdHandle, int algoNum, byte[] data, GrdAM method, byte[] iv)
```

grdHandle [in]

Тип: [Handle](#)

хэнгл, через который будет выполнена данная операция.

algNum [in]

Тип: int

Номер аппаратного алгоритма, с помощью которого будет происходить преобразование.

data [in]

Тип: byte []

Буфер данных для преобразования.

method [in]

Тип: [GrdAM](#)

Метод преобразования. Задается комбинацией флагов [GrdAM](#).

iv [in]

Тип: byte []

Вектор инициализации.

GrdE.AlgoNotFound	Алгоритм с указанным номером не существует
GrdE.CRCErrFunc	Ошибка CRC при выполнении алгоритма. Эта ошибка обычно возникает, если длина преобразуемой последовательности не совпадает с заданной во время создания алгоритма
GrdE.GPis0	Счетчик алгоритма достиг нулевого значения. Результат алгоритма больше нельзя получить

Метод **GrdTransform** позволяет преобразовывать информацию аппаратным алгоритмом ключа. Преобразование производится алгоритмом с порядковым номером, заданным в параметре *algNum*. Этот алгоритм предварительно должен быть создан. Если в дескрипторе алгоритма установлен флаг **nsaf1_GP_dec** (уменьшение счетчика), вычитание счетчика GP происходит при каждом вызове **GrdTransform**.

Если шифрование выполнялось блоками произвольной длины, то для корректного расшифрования длина и порядок обработки блоков должны сохраняться.

Для алгоритмов GSII64

Длина массива преобразуемых данных (в байтах) *data* зависит от режима работы алгоритма. Для режимов ECB и CBC длина данных должна быть кратной **GrdARS_GSII64** (8 байт), максимально **GrdAMRS_GSII64** (248 байт). Для режимов CFB и OFB длина может быть произвольной, но не превышающей 255 байт. Параметр *method* задается комбинацией флагов [GrdAM](#).

Массив данных для преобразования должен находиться по адресу, указанному в параметре *data*. Если метод выполнен успешно, по этому же адресу будет помещена последовательность преобразованных данных той же длины. В этом случае метод возвращает [GrdE.OK](#).

Скорость кодирования/декодирования напрямую зависит от длины блока данных *data*, передаваемого в **GrdTransform**. Максимальная скорость достигается при максимальной длине блока. Если размер блока данных сильно превышает максимальное значение, его нужно разбивать на куски максимально возможной длины. Однако при таком подходе ключ (особенно с интерфейсом LPT) может оказываться занятым на более долгое время в течении каждой такой операции. Поэтому в приложениях, для которых это критично (например, со множественными независимыми параллельными запросами к ключу), лучше использовать более мелкие блоки.

Для режимов, использующих сцепление блоков, необходимо задавать 8-байтовый вектор инициализации *iv*. При выполнении кодирования и декодирования необходимо задавать один и тот же вектор инициализации. Если кодирование/декодирование происходит в несколько приемов, то в качестве вектора инициализации нужно задавать значение, которое возвращается в *iv* после выполнения предыдущей операции.

Для алгоритмов AES128

Для работы с аппаратными алгоритмами AES128 используйте функцию [GrdTransformEx](#), т. к. только в этом случае есть возможность задания полного вектора инициализации (16 байт).

Для алгоритмов HASH64

Для работы с аппаратными алгоритмами HASH64 используйте функцию [GrdHash](#).

Для алгоритмов Guardant Stealth I и Guardant Fidus

Предупреждение

Данный тип алгоритмов не отвечает современным требованиям безопасности. Поддержка этих алгоритмов реализована исключительно в целях совместимости. В связи с этим алгоритмы данного типа не рекомендуются для использования в приложениях. Следует пользоваться алгоритмами GSII64, реализованными в Guardant Stealth/Net II/III.

Преобразование однонаправленное, т. е. для функции $F(X)$ не существует функция $F^{-1}()$ такая, что $X = F^{-1}(F(X))$. Оно производится алгоритмом с порядковым номером, заданным в параметре *algNum*. Этот алгоритм предварительно должен быть создан. Длина массива с преобразуемыми данными (в байтах) должна быть равной длине вопроса алгоритма, заданной в его дескрипторе (допустимые значения - от 4 до 248). Сам массив должен находиться по адресу, указанному в параметре *data*. Если метод выполнен успешно, по этому же адресу будет помещена последовательность преобразованных данных той же длины. В этом случае метод возвращает [GrdE.OK](#)