Guardant Code API. Интерфейс прикладного программирования загружаемого кода

При разработке загружаемого кода с большой вероятностью может возникнуть необходимость обращаться к ресурсам ключа, находящимся в области **EEPROM** (защищенным ячейкам, алгоритмам), или к таймеру. Поэтому был разработан специальный интерфейс прикладного программирования **Guardant Code API** (см. *Справочную систему Guardant API*) Библиотека этого API содержит большинство функций Guardant API, адаптированных для использования из среды загружаемого кода.

Основной нюанс при работе с **Guardant Code API** состоит в том, что хэндл защищенного контейнера в загружаемом коде теряет смысл, поскольку этот код, во-первых, имеет доступ только к одному ключу, а во-вторых, не существует ситуации конкурентного доступа к ресурсам ключа из разных потоков одного приложения и из разных приложений.

Вместе с тем, функциям внутреннего API загружаемого кода передается параметр типа HANDLE. Это сделано для соблюдения единообразия и удобства отладки загружаемого кода.
Guardant Code API поддерживает основные функции Guardant API, связанные с хранением данных и работой с алгоритмами.

Кроме того, в API загружаемого кода существует возможность вызывать криптографические алгоритмы, не используя дескрипторы, а напрямую, подобно тому, как в Guardant API вызываются программно-реализованные алгоритмы. Для этого вместо числового имени ячейки, содержащей дескриптор, указывается специальное зарезервированное имя алгоритма.

Если в загружаемом коде присутствуют функции Guardant API (например, в ключ переносится алгоритм, который раньше защищался ключами Guardant), то для большинства этих функций существуют аналоги в Guardant Code API, и портирование будет заключаться в смене префикса с **GrdXXX** на **GcaXXX** или **GccaXXX.**

Название функции	Краткое описание
GcaCrash()	Инициировать ошибку среды исполнения Guardant Code
GcaExit()	Выйти из загружаемого кода. В вызывающее приложение передается код возврата
GcaLedOn()	Включить светодиодный индикатор
GcaLedOff()	Выключить светодиодный индикатор
GcaRead()	Прочитать данные из EEPROM, аналогично GrdRead()
GcaWrite()	Записать данные в EEPROM, аналогично GrdWrite()
GcaPI_Read()	Прочитать данные из защищенной ячейки, аналогично GrdPI_Read()
GcaPI_Update()	Изменить данные защищенной ячейки, аналогично GrdPI_Update()
GcaPI_GetTimeLimit()	Получить время жизни защищенной ячейки, аналогично GrdPI_GetTimeLimit(). Для Guardant Code Time
GcaPI_GetCounter()	Получить значение счетчика оставшихся выполнений алгоритма, аналогично GrdPI_GetCounter()
GcaGetTime()	Получить текущее время из RTC ключа, аналогично GrdGetTime(). Для Guardant Code Time
GcaGetRTCQuality()	Проверить валидность значения RTC, аналогично GrdGetRTCQuality(). Для Guardant Code Time
GcaGetLastError()	Получить последний код ошибки, аналогично GrdGetLastError(). Возвращает код ошибки, которая произошла при последнем вызове функции Guardant Code API
GccaCryptEx()	Шифровать данные симметричным алгоритмом, аналогично GrdCryptEx(). Для программного алгоритма AES128 требуется другой размер контекста. Контекст должен быть выровнен по границе в 4 байта (!) применением макроса ALIGNED
GccaSign()	Вычислить ЭЦП, аналогично GrdSign(). Присутствует дополнительный параметр – ключ подписи и вариант вызова, работающий в обход таблицы дескрипторов
GccaVerifySign()	Проверить ЭЦП, аналогично GrdVerifySign()

GccaGenerateKeyPair()	Генерировать ключевую пару для алгоритма ЭЦП ECC160
GccaHash()	Вычислить хэш-функцию, аналогично GrdHash()
GccaGetRandom()	Генерировать случайное однобайтовое число
GcaSetTimeout	Установить разрешенное время работы загружаемого кода
GcaCodeGetInfo	Запросить информацию из дескриптора загружаемого кода
GcaCodeRun	Выполнить код из другого участка загружаемого кода

Важно!

1. Подробную информацию по функциям внутреннего **Guardant Code API** см. в Справочной системе по Guardant API
2. Поскольку в Guardant Code не реализован алгоритм GSII64 и производные от него (HASH64, RAND64 и т. д.), возможно придется немного переработать существующую схему защиты на использование алгоритмов AES128 для шифрования и SHA256 для хэширования. Все остальные возможности предыдущих поколений ключей присутствуют и в Guardant Code.

Сервисные функции GrdAPI для работы с загружаемым кодом

Загружаемый код, в отличие от ячеек с данными и дескрипторов аппаратных алгоритмов, хранится в другой области памяти ключа, для которой используются иные принципы адресации. Поэтому для загрузки кода в память ключа, его выполнения и других операций существуют специальные функции Guardant API.

Для записи в ключ загружаемый код преобразуется в файл специального формата **GCEXE** (Guardant Code Executable), обеспечивающего защиту от подделки, подмены и анализа. Эта защита обеспечивается шифрованием криптостойкими алгоритмами и ЭЦП. Такая защита делает возможным безопасное обновление загружаемого кода в ключе, в том числе при пересылке файлов по открытым каналам и через сети общего пользования.

Файл формата **GCEXE** генерируется на основе данных дескриптора загружаемого кода, скомпилированного кода и map-файла утилитой программирования ключей **GrdUtil.exe**.

Однажды сгенерированный файл может быть использован для записи в тиражируемые ключи той же утилитой. Если предпродажное программирование осуществляется при помощи собственных специально разработанных инструментов, файл с загружаемым кодом может быть записан в ключ при помощи функции GrdCodeLoad().

Список сервисных функций Guardant API:

Название функции	Код доступа	Краткое описание
GrdCodeGetInfo	Private Read	Получить информацию из дескриптора загружаемого кода
GrdCodeLoad	Private Read	Записать GCEXE-файл, содержащий загружаемый код, во Flash-память ключа
GrdCodeRun	Private Read	Выполнить загружаемый код
GrdSetDriverMode	Private Read	Задать USB-режим работы ключа