Особенности Guardant API

Псевдокод

Псевдокод - это технология защиты исполняемых файлов от изучения логики их работы. Суть ее состоит в том, что определенные фрагменты исполняемых файлов дизассемблируются, анализируются и преобразуются в защищенный код виртуальной машины, которая генерируется тут же.
Анализировать логику работы защищенного подобным образом кода на порядок сложнее, чем инструкции Intel-совместимых процессоров, поскольку для псевдокода не существует никакого стандартного инструментария (отладчиков, дизассемблеров). Взломщику придется все делать вручную или создавать свои собственные инструменты.
В каждой копии виртуальной машины реализуется уникальным образом:

- Набор внутренних команд псевдокода
- Множественный взаимный контроль целостности(для затруднения модификации и установки точек останова)
- Обфускация кода виртуальной машины(замусоривание реального кода вторичным)
- Преобразование кода самой виртуальной машины
- Преобразование самого псевдокода
- Параметры многих команд рассчитываются только во время выполнения (защита от статического дизассемблирования)
- Отсутствие постоянных сигнатур в защищенном коде (затруднение поиска псевдокода в защищенном приложении)

Handle-ориентированность

Для работы с ключом создается контекст, который сохраняетсяв специальном защищенном контейнере. Доступ к контейнеру осуществляется через хэндл. Хэндл является идентификатором контекста, с которым и оперируют функции Guardant API.
Это позволяет упростить обмен с API за счет минимизации числа вызовов и параметров функций, а также сократить время работы функций.

Поддержка многопоточности

В защищенном контейнере сохраняются параметры поиска и прочие внутренние переменные. Это существенно расширяет возможности разработчиков приложений, поскольку наличие контекста позволяет работать с несколькими ключами из нескольких потоков одновременно.
Это дает возможность:

- Существенно повысить защищенность, поскольку отлаживать и анализировать многопоточные приложения гораздо сложнее
- Легче интегрировать API в сложные многопоточныеприложения
- Работать с несколькими ключами одновременно

Защищенность, контроль целостностии аутентификация

Содержимое контейнера закодировано. Осуществляется контроль целостности данных, хранящихся в контейнере. Коды доступа тоже хранятся в контейнере в закодированном виде и их не нужно указывать при каждом вызове функций Guardant API. В защищенном контейнере хранятся только те коды доступа, которые действительно используются.
Guardant API контролирует целостность своего кода во избежание вмешательств извне. Код Guardant API защищен от статического и динамического анализа при помощи псевдокода.
Драйвер Guardant подписан при помощи асимметричной криптографии. Защищенное приложение проверяет подпись драйвера (автоматически при вызове функций Guardant API 5.x). Взломщик не сможет создать драйвер-эмулятор электронного ключа, чтобы у него сходилась подпись, поскольку закрытый ключ ему не известен.
Так же как и код Guardant API, драйвер защищен от анализа с использованием технологии псевдокода. Это подразумевает, в том числе, и контроль целостности кода драйвера.

Унификация

Для унификации и упрощения написания кода все функции(методы) Guardant API сделаны универсальными для работы, как с локальными, так и с сетевыми ключами. При инициализации библиотеки необходимо указать, с какими ключами будет работать приложение: локальными, сетевыми, или с обоими типами.
Это существенно упрощает процесс программирования систем защиты, рассчитанных на работу и с сетевыми и с локальными ключами, поскольку код защиты можно написать единый. Также максимально упрощается переход на сетевые ключи: по сути надо будет поменять всего один флаг в функции(методе) **GrdSetFindMode()**.