Sentinel LDK

Sentinel LDK v.7.4

Документация

SafeNet

Оглавление

1. О комплекте разработчика Sentinel LDK 3

2. Описание инструментов Sentinel LDK 6

2.1. Sentinel Vendor Suite 6

2.2. Sentinel LDK EMS 7

2.3. Sentinel LDK Envelope 8

2.4. Sentinel LDK Toolbox 9

2.5. Sentinel Master Wizard 10

2.6. Sentinel API 11

2.7. Sentinel Admin Control Center 12

3. Порядок работы с системой защиты Sentinel LDK 14

3.1. Установка 14

3.2. Процедура представления служебных ключей 17

3.3. Защита приложений 21

3.3.1. С помощью Sentinel LDK Envelope 21

3.3.2. С помощью Licensing API 26

3.4. Запись лицензий в ключи защиты с использованием License Generation API 30

3.5. Запись лицензий в ключи защиты с использованием Sentinel LDK EMS 33

4. Официально поддерживаемые платформы 33

5. Контакты технической поддержки 39

1. О комплекте разработчика Sentinel LDK

Комплект разработчика Sentinel LDK представляет собой набор утилит, позволяющих реализовать работу клиентского приложения с аппаратным ключом защиты Sentinel HL, либо с программным ключом защиты Sentinel SL, записывать лицензии в клиентские ключи Sentinel HL / Sentinel SL, а также производить удалённое обновление лицензий в клиентских ключах защиты.

Комплект разработчика Sentinel LDK решает следующие задачи:

- защита готовых программ (*.exe, *.dll, *.jar, *.war, *.apk, *.so, ELF файлы);
- гибкая защита программного обеспечения при помощи Sentinel LDK API;
- лицензирование и защита отдельных модулей и функций ПО;
- защита контента (файлов: *.flv, *.swf и т.д.);
- программная защита для продажи и активации программ через Интернет (ключи Sentinel SL).

Ключи защиты подразделяются на:

1) Служебные ключи – это ключи разработчика ПО, необходимые для защиты приложений, а также для записи лицензий в пользовательские ключи защиты.
   Служебные ключи бывают двух видов:
   - **Sentinel HL Master Key**
     Ключ синего цвета, на этикетке ключа написано «MASTER», содержит уникальные коды и идентификаторы, используемые ключами защиты Sentinel HL, которые присваиваются разработчику компанией SafeNet. Используется для записи лицензий в USB-ключи Sentinel HL.
   - **Sentinel HL Developer Key**
     Ключ жёлтого цвета, на этикетке ключа написано «DEVELOPER», содержит уникальные коды и идентификаторы, используемые ключами защиты Sentinel HL, которые присваиваются разработчику компанией Gemalto. Используется при построении защиты программ с помощью USB ключей Sentinel HL.

2) Пользовательские ключи – ключи, на которые осуществляется защита приложения, в свою очередь они разделяются на программные ключи Sentinel SL и аппаратные ключи Sentinel HL.

   Программные ключи Sentinel SL и аппаратные ключи Sentinel HL по принципу работы никак друг от друга не отличаются, и защищённому приложению всё равно, с какими из этих ключей работать.

   В свою очередь программные ключи Sentinel SL бывают нескольких типов:

   - **SL-AdminMode**
     Ключи защиты устанавливаются на ПК, где уже установлен драйвер для ключей Sentinel;
   - **SL-UserMode**
     Ключи защиты устанавливаются на ПК, где нет установленного драйвера для ключей Sentinel. Защищённое приложение с такими ключами работает без драйвера, используя для коммуникации с ключом портативный менеджер лицензий hasp_rt.exe (см. раздел о защите приложений 3.3.1. С помощью Sentinel LDK Envelope);

- **Trial Ware (Provisional SL)**
  Локальный триальный ключ защиты, выпускаемый на срок от 1 до 90 дней, без привязки к какому-либо конкретному компьютеру. Генерируется V2C файл с таким ключом, который может устанавливаться на любой компьютер и будет работать там в течение того срока, который разработчик ПО определил в момент создания данного ключа. При этом счётчик времени начнёт свой отчёт с момента первого использования ключа (когда защищённое ПО впервые обратится к ключу за лицензией). Лицензия на выпуск таких ключей для каждого Мастер-ключа приобретается отдельно.
- **Unlocked License (Perpetual Provisional SL)**
  Локальный триальный ключ защиты без ограничений по времени. Лицензия на выпуск таких ключей для каждого Мастер-ключа приобретается отдельно.

Аппаратные ключ Sentinel HL подразделяются по моделям:

- **Sentinel HL Basic**
  Наиболее простое и эффективное решение для защиты недорогих программ, не требующих управления лицензированием и сохранения в памяти ключа защиты параметров и настроек.

  Алгоритм: AES · Уникальный ID: нет · Память RW/R: нет · Кол-во лицензий: 1

- **Sentinel HL Pro**
  Самая подходящая модель для защиты программного обеспечения с лицензированием по используемым компонентам. Наиболее популярные USB-ключи для защиты программ и данных.

  Алгоритм: AES · Уникальный ID: да · Память RW/R: 112/112 байт · Кол-во лицензий: 11-39

- **Sentinel HL Max**
  Оптимален для защиты сложного программного обеспечения с лицензированием по функциональности и/или количественным показателям.

  Алгоритм: AES · Уникальный ID: да · Память RW/R/D*: 4/2/25 КБ · Кол-во лицензий: 240-2160

- **Sentinel HL Max micro**
  Благодаря миниатюрному корпусу практически не выступает из USB-порта, очень удобен при использовании в ноутбуках и планшетах. По функционалу аналогичен ключу Sentinel HL Max.

  Алгоритм: AES · Уникальный ID: да · Память RW/R/D*: 4/2/25 КБ · Кол-во лицензий: 240-2160

- **Sentinel HL Time**
  Эффективное решение для организации аренды, лизинга, подписки на защищенное программное обеспечение, распространения пробных версий (trial). Аналогичен модели Sentinel HL Max, но дополнительно имеет часы реального времени, используемые при лицензировании программного обеспечения.

Алгоритм: AES · Уникальный ID: да · Память RW/R/D*: 4/2/25 КБ · Кол-во лицензий: 240-2160

- **Sentinel HL Net**
  Сетевые модели, разработанные специально для защиты корпоративного ПО, при этом USB-ключи могут работать и как локальные. Ключ защиты Sentinel HL Net позволяет ограничивать количество пользователей, одновременно работающих с защищенными программами, лицензировать компоненты, функциональность и другие количественные показатели.

  Алгоритм: AES · Уникальный ID: да · Память RW/R/D*: 4/2/25 КБ · Кол-во лицензий: 240-2160

- **Sentinel HL NetTime**
  Совмещает в себе функции ключей Sentinel HL Net и Sentinel HL Time, оптимален для ограничения работы по времени защищенных программ совместно с контролем количества одновременно работающих пользователей.

  Алгоритм: AES · Уникальный ID: да · Память RW/R/D*: 4/2/25 КБ · Кол-во лицензий: 240-2160

- **Sentinel HL Drive**
  Предназначен для распространения программного обеспечения на ключе со встроенной Flash-памятью. Совмещает в себе функции ключа Sentinel HL Max и Flash-накопителя.

  Алгоритм: AES · Уникальный ID: да · Память RW/R/D*: 4/2/25 КБ · Кол-во лицензий: 240-2160

- **Sentinel HL Max Board**
  Модель предназначена для производителей Embedded систем из готовых комплектующих. Sentinel HL Max Board монтируется непосредственно во внутренний USB разъём материнской платы, что позволяет скрыть ключ внутри корпуса Embedded устройств. По функционалу аналогичен ключу Sentinel HL Max.

  Алгоритм: AES · Уникальный ID: да · Память RW/R/D*: 4/2/25 КБ · Кол-во лицензий: 240-2160

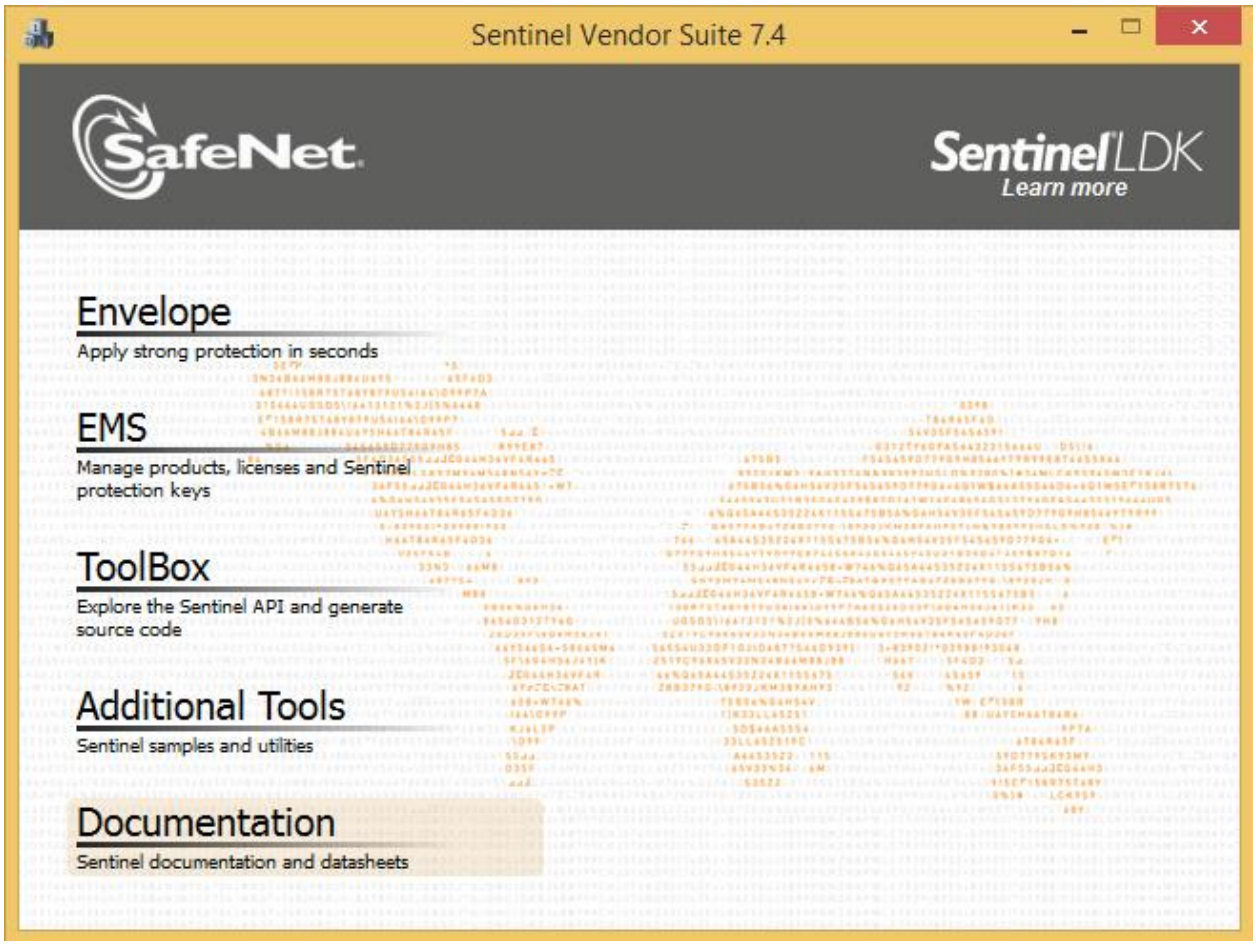
- **Sentinel HL Max Chip**
  Модель предназначена для разработчиков ПО, имеющих производство собственных комплектующих. Ключ имеет стандартный интерфейс USB-шины(SOIC8) для припайки на плату. По функционалу аналогичен ключу Sentinel HL Max.

  Алгоритм: AES · Уникальный ID: да · Память RW/R/D*: 4/2/25 КБ · Кол-во лицензий: 240-2160

** RW – память предназначенная для чтения и записи, R – только для чтения, D – динамически распределяемая память, может использоваться для обоих вариантов, указанных выше.*

2. Описание инструментов Sentinel LDK

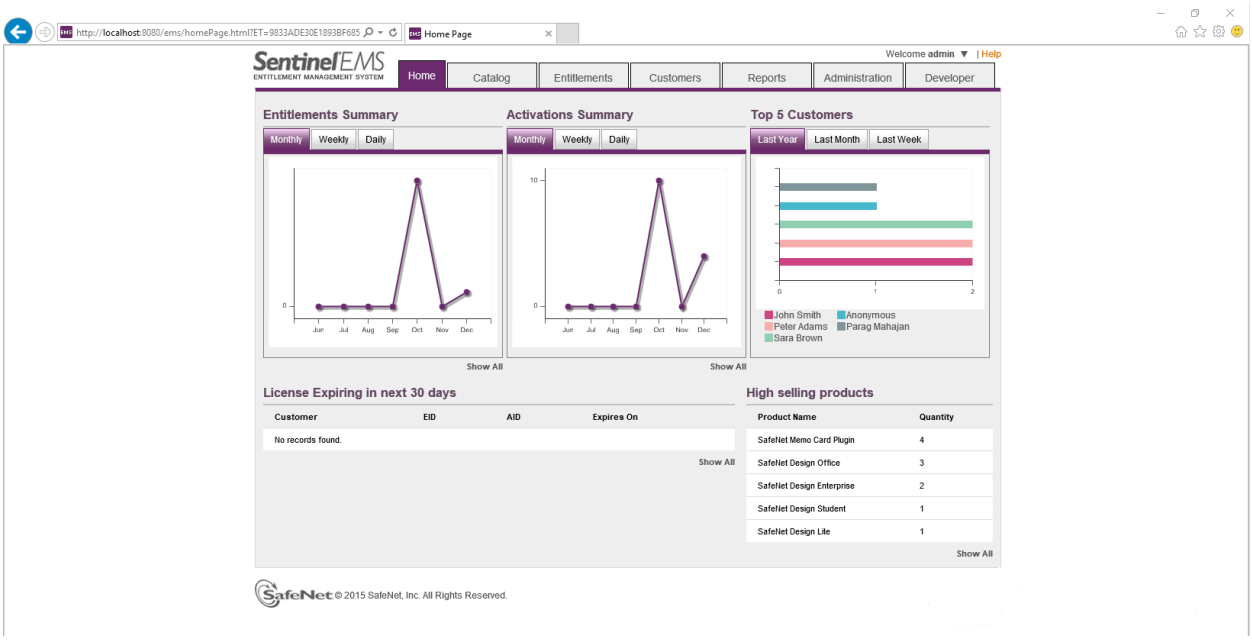
2.1. Sentinel Vendor Suite



Это общий интерфейс, служащий для запуска всех остальных компонент из комплекта разработчика:

- Sentinel LDK EMS
- Sentinel LDK Envelope;
- Sentinel LDK Toolbox;
- Sentinel Master Wizard;
- Sentinel API;
- Sentinel Admin Control Center и т.д.

2.2. Sentinel LDK EMS



Сервис Sentinel LDK EMS для управления лицензиями. Он позволяет одинаково удобно формировать лицензии на многофункциональные программные пакеты и на небольшие программы узкого назначения. С его помощью в любой момент можно изменить лицензионную политику для своих продуктов в соответствии с требованиями рынка и заказчиков.

Функции Sentinel LDK EMS

Запись лицензий в USB-ключи

Для записи (прошивки) лицензии в ключ достаточно иметь возможность подключиться с помощью веб-браузера к серверу с Sentinel LDK EMS и иметь права, позволяющие записывать лицензии. Процесс записи лицензий не зависит от установленной операционной системы, главное – иметь возможность подключить ключ к USB-порту.

Активация программных ключей

Онлайн активация. Модуль активации (который может быть встроен в защищаемое ПО) выполняет прямое соединение с сервером Sentinel LDK EMS для обмена служебной информацией. Пользователю достаточно ввести ключ активации, чтобы ПО было активировано автоматически.

Портал активации. Пользователь может самостоятельно перейти на портал активации, указать свой ключ и выполнить активацию ПО. Опционально, сервер Sentinel LDK EMS может собирать регистрационные данные пользователей.

Офлайн активация. Если пользователь не имеет прямого выхода в Интернет, есть возможность выполнить активацию, обменявшись служебной информацией с разработчиком по электронной почте.

Обновление лицензий

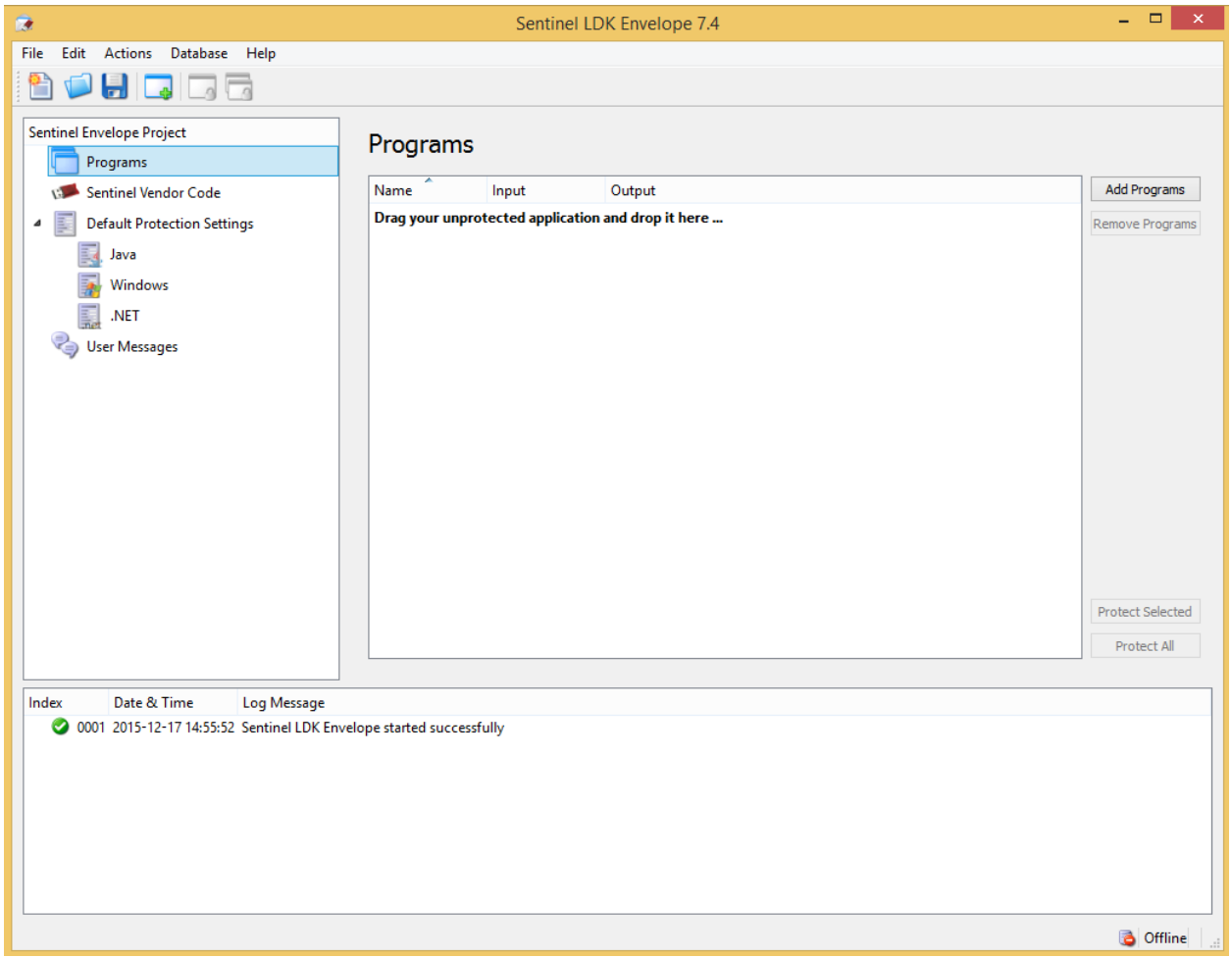
Если необходимо изменить набор лицензий в программном обеспечении, уже установленном у пользователя, Sentinel LDK EMS позволяет это делать онлайн (для программных ключей), либо удалённо по электронной почте.

Демо-версии

Sentinel LDK EMS позволяет создавать лицензии (Provisional license), не требующие активации. Такая лицензия может быть размещена на сайте. Она будет копироваться пользователями без каких-либо ограничений, но не может быть установлена повторно на одну и ту же машину. Можно выбрать один из двух режимов работы:

- от 1 до 90 дней с момента первого запуска ПО – Trial Ware (Provisional SL);
- постоянная лицензия – Unlocked License (Perpetual Provisional SL).

2.3. Sentinel LDK Envelope



Утилита Sentinel LDK Envelope предназначена для автоматической защиты уже скомпилированных приложений. При этом она даёт возможность устанавливать различные настройки защиты, тем самым делая её более гибкой.

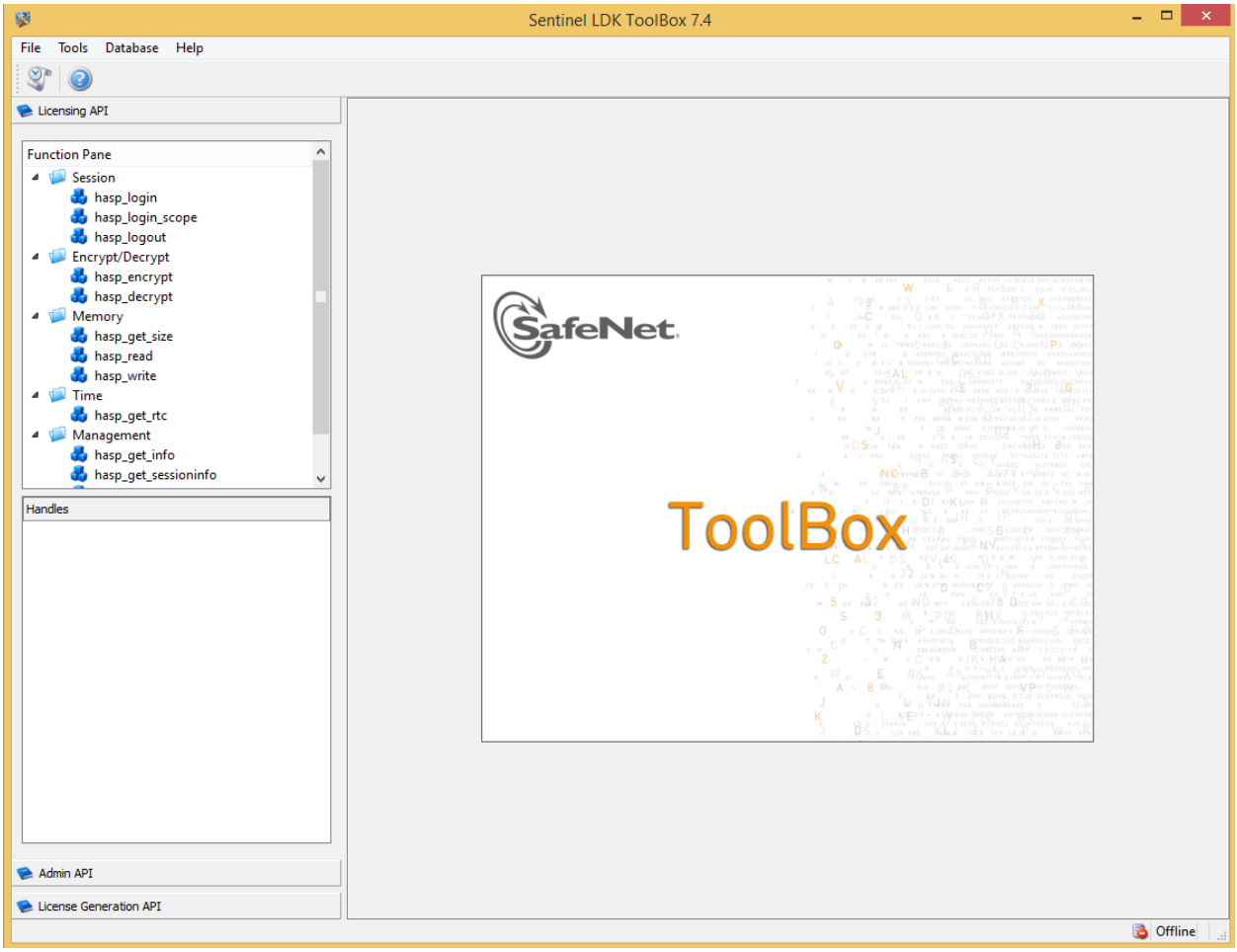
Утилита Envelope существует под различные операционные системы:

- Windows – позволяет защищать: *.exe, *.dll, *.jar, *.war, *.apk файлы;
- Linux – позволяет защищать: *.so, ELF файлы;
- Mac OS – позволяет защищать: Mach-O файлы.

Реализация защиты Sentinel LDK Envelope является быстрым способом построения надёжной защиты программного обеспечения и не требует внесения изменений в исходный код самого защищаемого приложения. Графический интерфейс Sentinel LDK Envelope позволяет легко

изменять параметры защиты для защищаемых файлов, а также настраивать сообщения об ошибках, которые отображаются при работе пользователей с защищенным приложением.

2.4. Sentinel LDK Toolbox



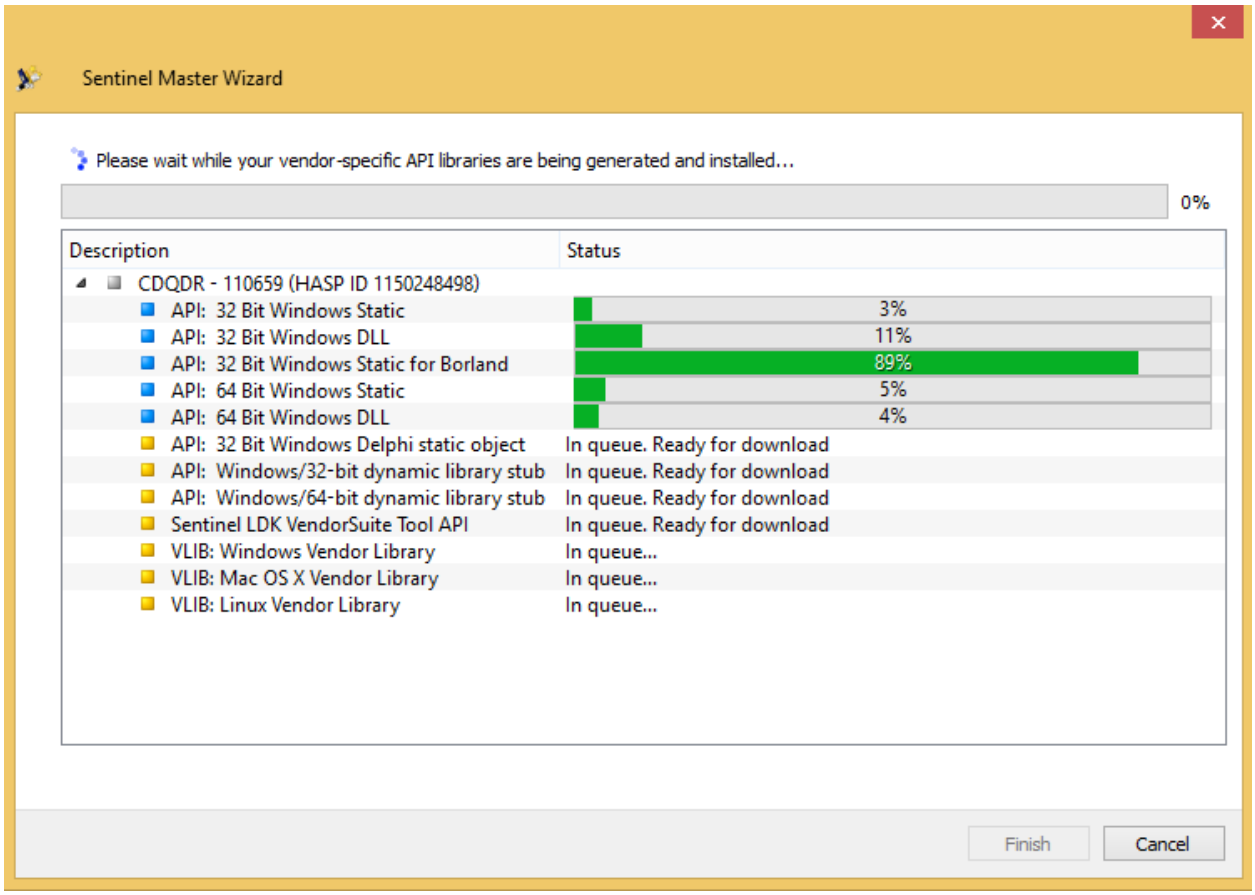
Утилита Sentinel LDK Toolbox предназначена для демонстрации работы с API функциями. Имеет возможность генерации кода вызова API функций под различные языки программирования (C, C#, C++, Java, VB.NET).

Sentinel LDK Toolbox позволяет работать с тремя API:

1) Licensing API (оно же Runtime API) – предназначено для работы с ключом защиты.
2) Admin API – предназначено для работы с менеджером лицензий (Admin Control Center, ACC).
3) License Generation API (оно же Licgen API) – предназначено для записи лицензий в ключ без использования Sentinel EMS.

Sentinel LDK Toolbox существует только под операционную систему Windows.

2.5. Sentinel Master Wizard



Утилита Sentinel Master Wizard предназначена пря выполнения процедуры представления служебного (Мастер или Developer) ключа. Перед началом работы с ключами каждой серии разработчика необходимо провести процедуру представления служебного ключа. В ходе этой процедуры происходит загрузка Vendor кода, а также кастомизированных API библиотек под используемую серию ключей на компьютер клиента.

Sentinel Master Wizard существует под различные операционные системы:

- Windows;
- Linux;
- Mac OS.

Кастомизированные API библиотеки для каждой операционной системы необходимо получать, выполняя процедуру представления служебного ключа именно под данной операционной системой, используя соответствующую утилиту Sentinel Master Wizard.

Для работы утилиты Sentinel Master Wizard необходимо:

- Запускать утилиту Sentinel Master Wizard от имени администратора;
- Иметь подключенный служебный ключ: Мастер или Developer ключ;
- Иметь прямой доступ к Интернету (без промежуточных прокси серверов), либо иметь MWP файл для выполнения процедуры в офлайн режиме.

MWP для выполнения процедуры в офлайн режиме запрашивается отдельно у технической поддержки.

Процедура запроса MWP файла:

На адрес support-ru@safenet-inc.com необходимо написать письмо вида:

Тема письма:

Запрос MWP файла для компании *Название компании*

Тело письма:

1) Batch code – код разработчика (серия разработчика), для которого требуется MWP файл. На этикетке каждого ключа Sentinel HL – и служебного, и рабочего – нанесены дата продажи, модель, версия прошивки (firmware) ключа и пять латинских букв – код разработчика.

2) Версию комплекта разработчика, который Вы намереваетесь использовать (достаточно указать версию Vendor Suite).

3) Тип используемой ОС:
   - Windows;
   - Linux;
   - Mac OS.

2.6. Sentinel API

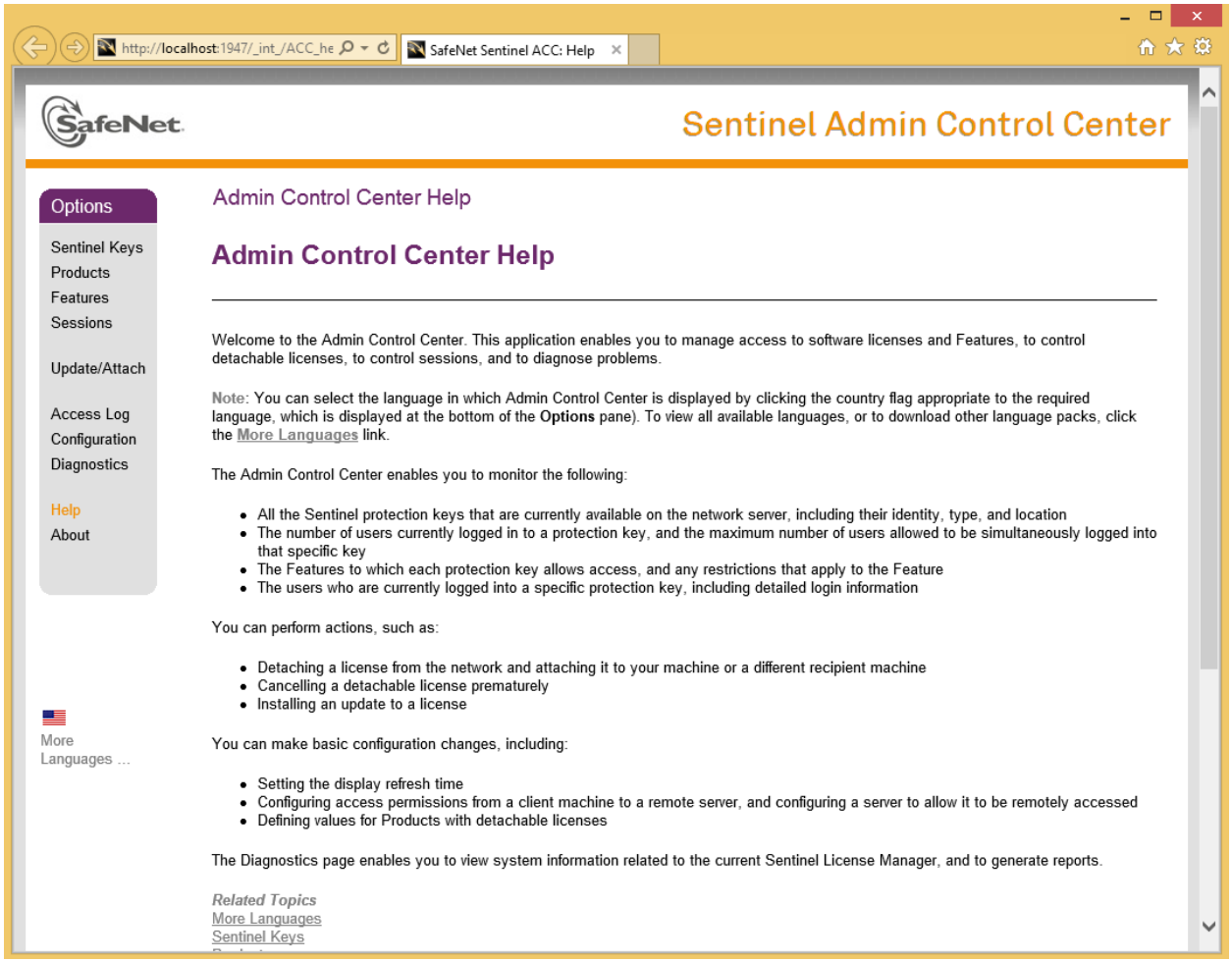
В комплекте разработчика представлены следующие виды API:

1) Licensing API (Runtime API) – предназначено для работы с ключом защиты, позволяет выполнять проверку наличия ключа защиты на ПК, а также наличие различных лицензий, записанных в ключ защиты; создавать сессии с ключом защиты; читать/писать из/в память ключа в рамках установленной сессии с ключом; использовать криптопроцессор ключа для шифрования/расшифровывания данных и т.д.
2) Admin API – предназначено для работы с менеджером лицензий (Admin Control Center), позволяет получать информацию обо всех лицензиях, расположенных на подключенных к ПК или доступных по сети ключах защиты.
3) License Generation API (Licgen API) – предназначено для записи лицензий в ключ защиты, а также для записи данных в память ключа.

Кастомизированным является только Licensing API, используемое для построения защиты приложений, остальные API являются общими для всех серий разработчика.

До выполнения процедуры представления служебного ключа на ПК с установленным комплектом разработчика кастомизированное API и примеры работы с ним доступны только для демонстрационного кода разработчика DEMOMA, интегрированного в сам комплект разработчика и не требующего для своей работы служебных ключей. Серия DEMOMA используется для демонстрации возможностей системы защиты Sentinel LDK.

2.7. Sentinel Admin Control Center



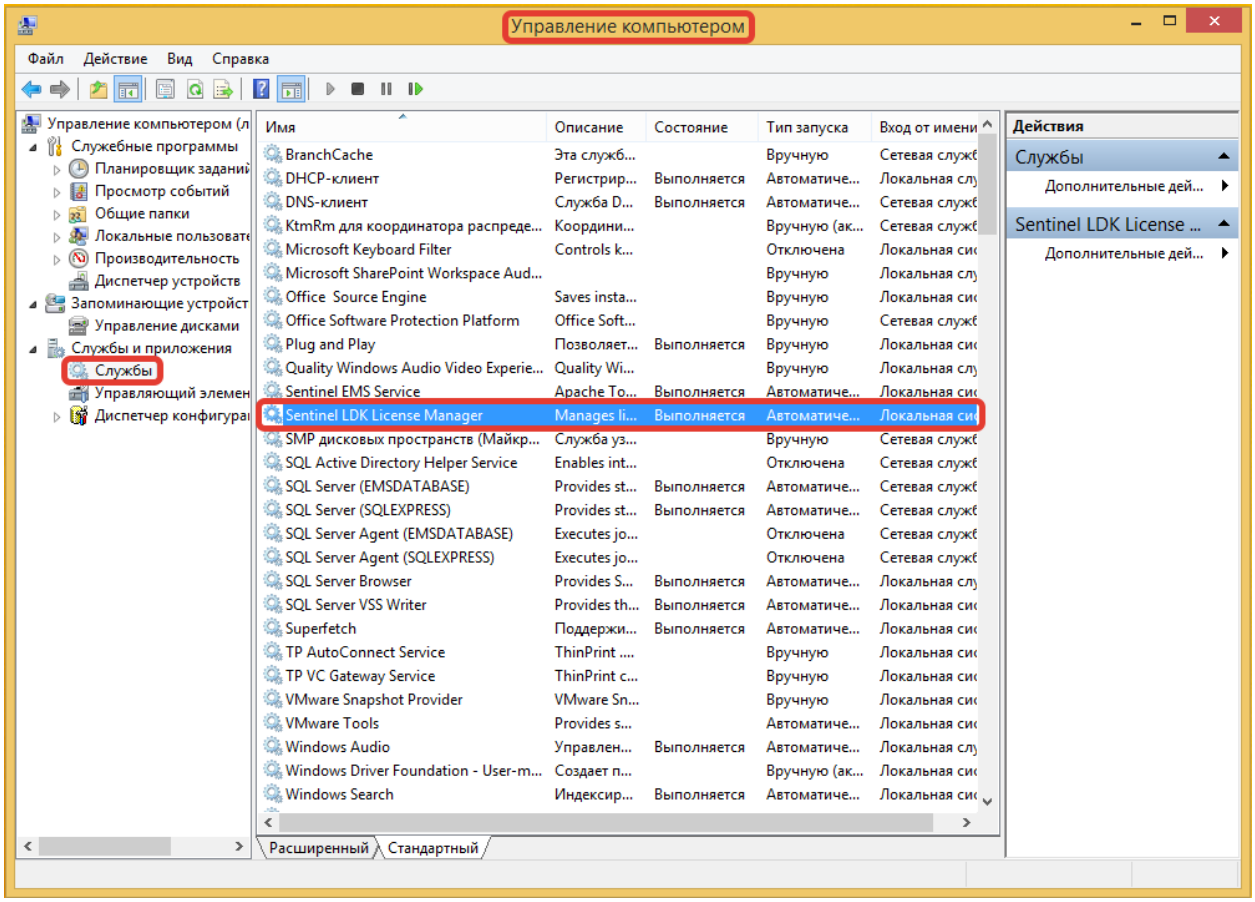
Утилита Sentinel Admin Control Center представляет собой менеджер лицензий, встроенный в драйвер ключа защиты, имеет веб-интерфейс, доступный через браузер на любом ПК, где установлен драйвер, по адресу http://localhost:1947

В интерфейсе Sentinel Admin Control Center отображается информация обо всех установленных на ПК или доступных по сети ключах защиты, информация о лицензиях в ключах, созданных (в ходе работы защищённого приложения) сессиях с ключом защиты, а также о текущих настройках менеджера лицензий и о текущих используемых компонентах драйвера.

Sentinel Admin Control Center может использоваться для применения обновлений к ключам защиты, выполнения временного переноса части лицензий из программного ключа защиты и т.д.

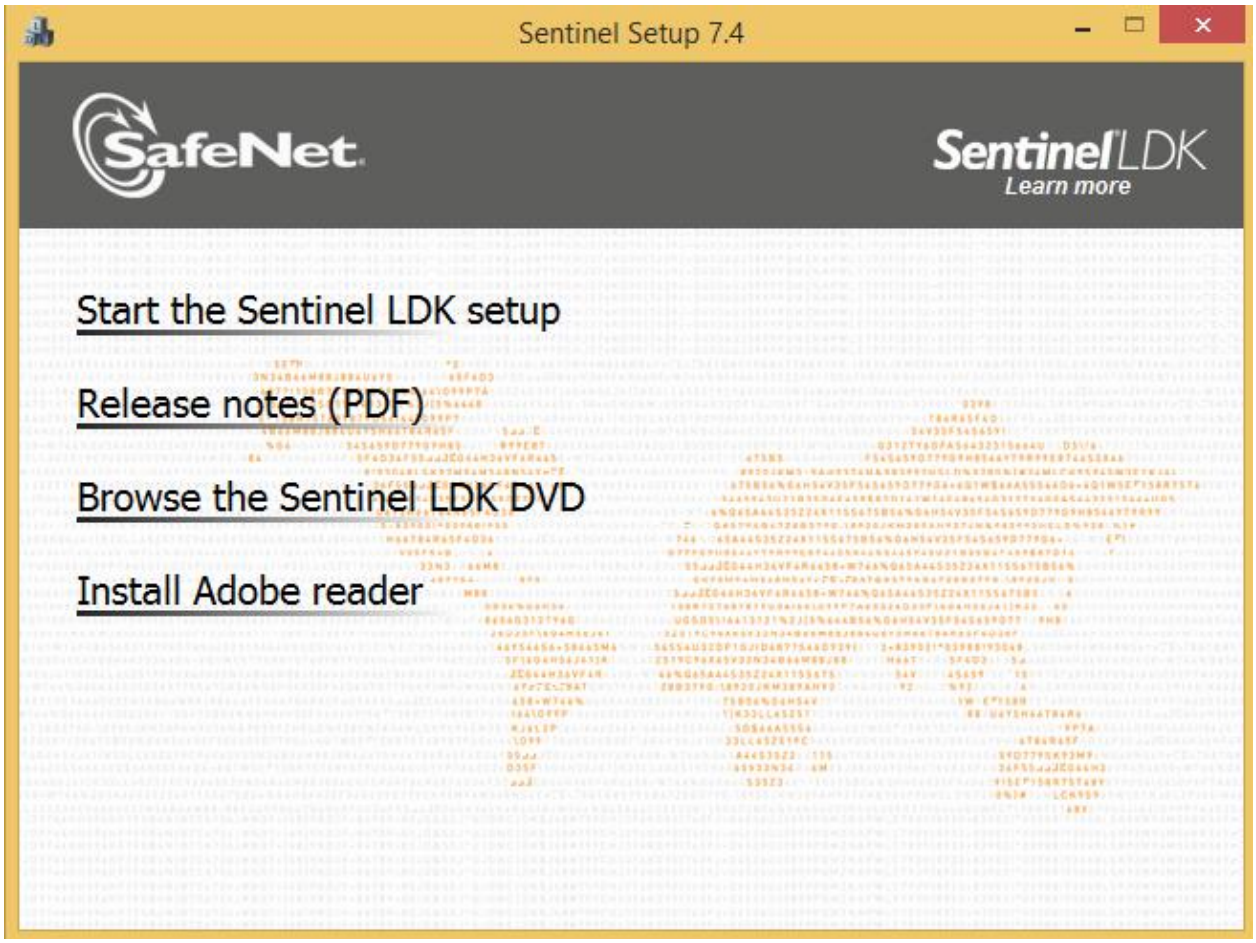
Sentinel Admin Control Center для своей работы использует порт 1947 и обменивается данными с другими менеджерами в сети, используя протоколы TCP и UDP.

За работу Sentinel Admin Control Center отвечает служба Sentinel LDK License Manager (процесс hasplms.exe):



3. Порядок работы с системой защиты Sentinel LDK

3.1. Установка

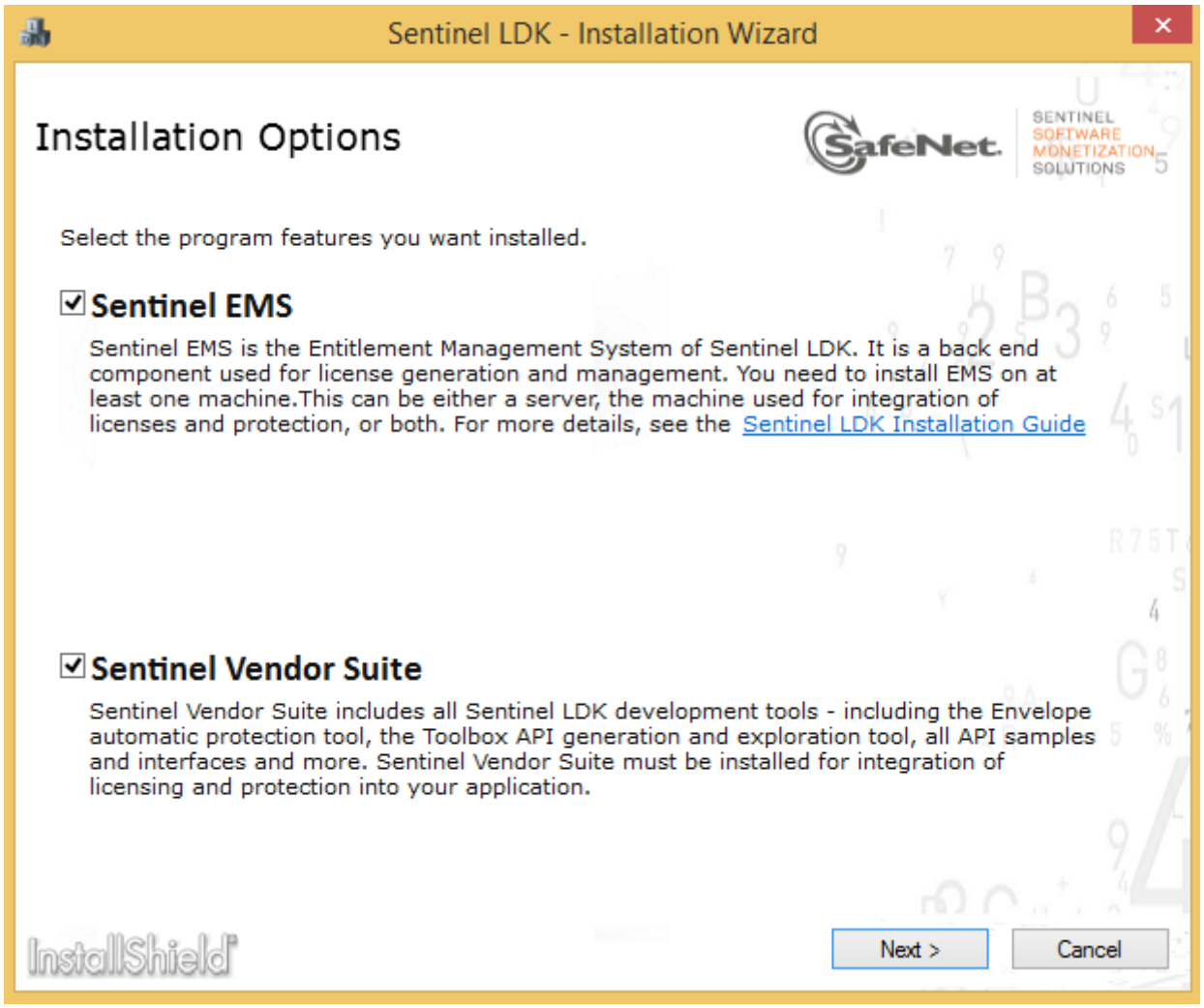


Установку комплекта разработчика Sentinel LDK лучше выполнять от имени локального администратора.

Установка как таковая осуществляется только под Windows, под Linux и Mac OS достаточно скопировать содержимое соответствующих директорий из дистрибутива на ПК, и можно запускать утилиты из комплекта разработчика:

- Linux: *“*Дистрибутив с комплектом разработчика**\Linux\VendorTools\”*
- Mac OS: *“*Дистрибутив с комплектом разработчика**\MacOS\VendorTools\VendorSuite\”*

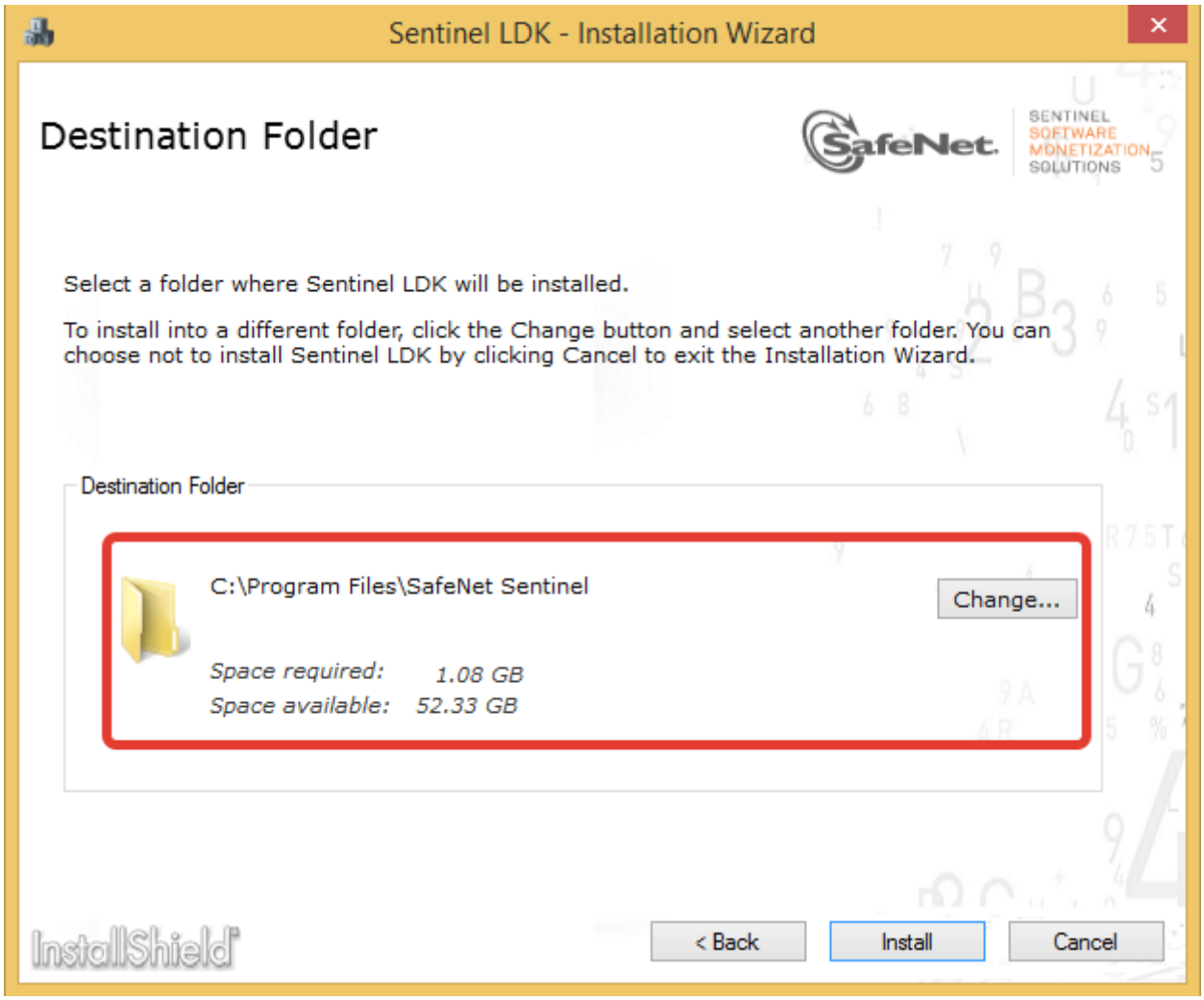
При установке комплекта разработчика Sentinel LDK под Windows инсталлятор позволяет выбрать устанавливаемые компоненты:



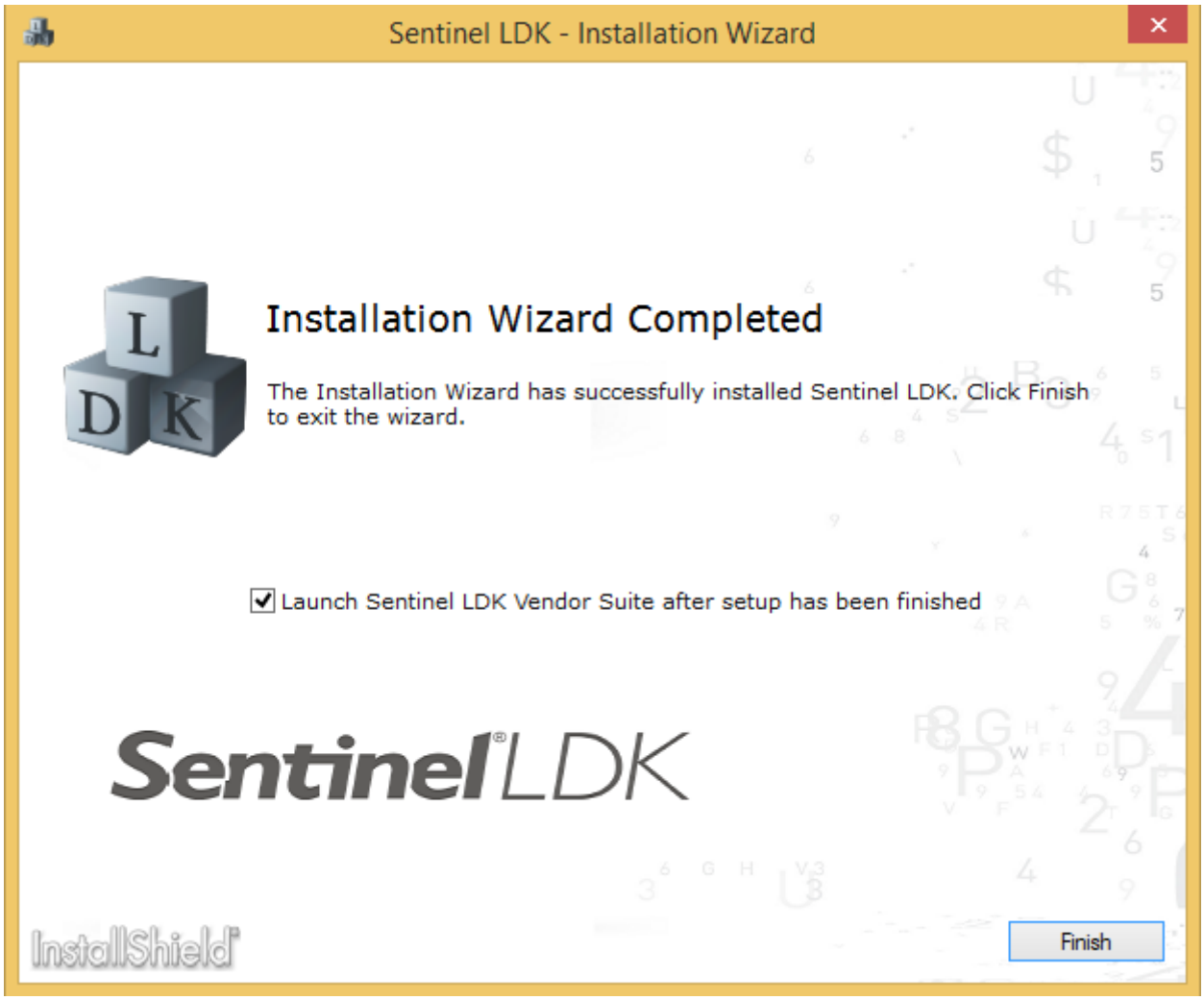
1) Sentinel EMS – система лицензирования, позволяющая записывать лицензии в пользовательские ключи, вести базу данных клиентов, их заказов, лицензий и продуктов.
2) Sentinel Vendor Suite – набор утилит, объединяющих все остальные инструменты из комплекта разработчика (Sentinel Envelope, Sentinel Toolbox, API, и т.д.).

По умолчанию выбраны оба компонента, однако установка Sentinel EMS не обязательна при использовании License Generation API для записи лицензий в пользовательские ключи защиты.

Помимо этого, установщик позволяет выбрать директорию установки комплекта разработчика:



Затем производится установка комплекта разработчика, а именно установка драйвера для ключей защиты Sentinel, и после этого установка выбранных ранее компонент комплекта разработчика.



3.2. Процедура представления служебных ключей

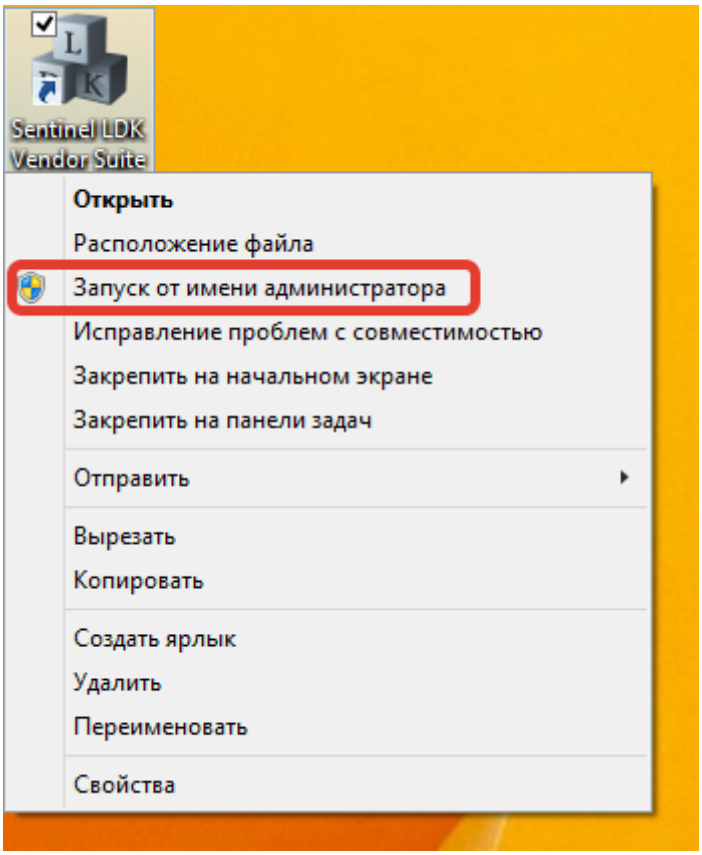
После установки комплекта разработчика на ПК необходимо выполнить обязательную процедуру представления служебного ключа (лучше использовать для этого Мастер ключ).

Данная процедура необходима для получения кастомизированных API библиотек и файла с Vendor кодом, необходимых для работы с ключами каждой серии разработчика.

Кастомизированные API библиотеки и файл с Vendor кодом для каждой серии разработчика свои, соответственно для каждой серии разработчика в начале работы с комплектом разработчика Sentinel LDK необходимо выполнять данную процедуру.

Для её выполнения необходимо:

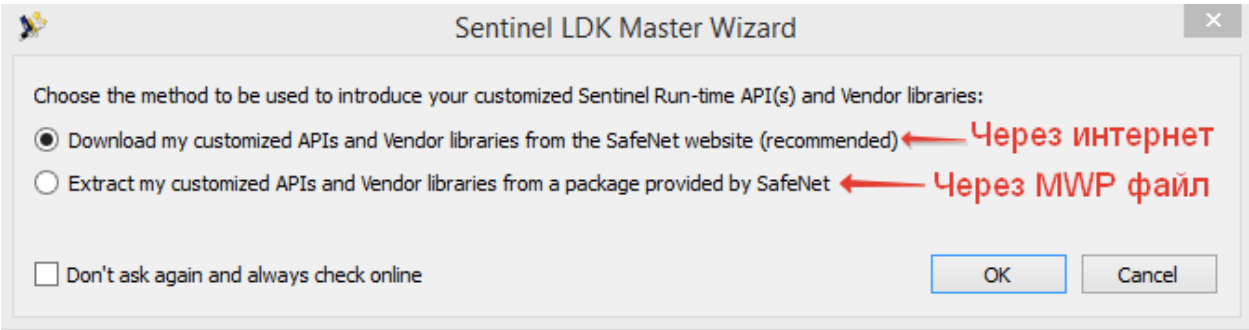
1) Подключить к ПК служебный ключ (Мастер или Developer);
2) Подключить на ПК прямой доступ в интернет (желательно без промежуточных прокси-серверов), либо запросить MWP файл для офлайн процедуры представления служебных ключей;
3) Запустить утилиту Sentinel LDK Vendor Suite от имени администратора:



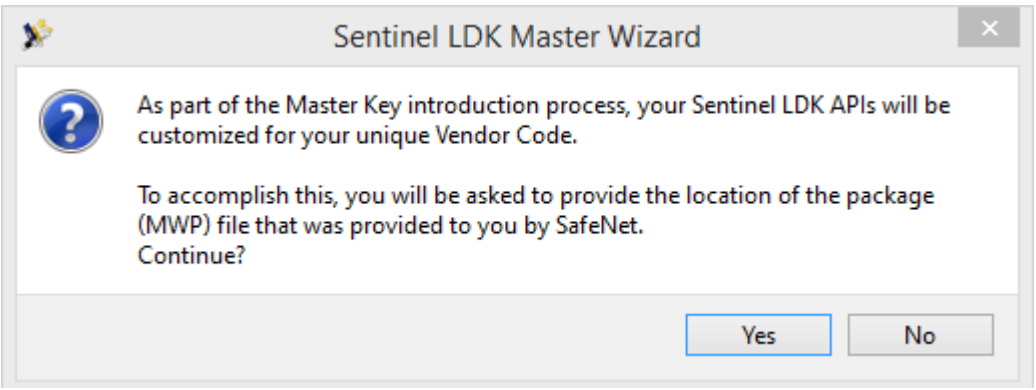
Из него запустить утилиту Sentinel Master Wizard, таким образом, утилита также будет запущена от имени администратора.

При наличии Интернета появится окно с выбором варианта проведения процедуры представления служебного ключа, всего вариантов два:

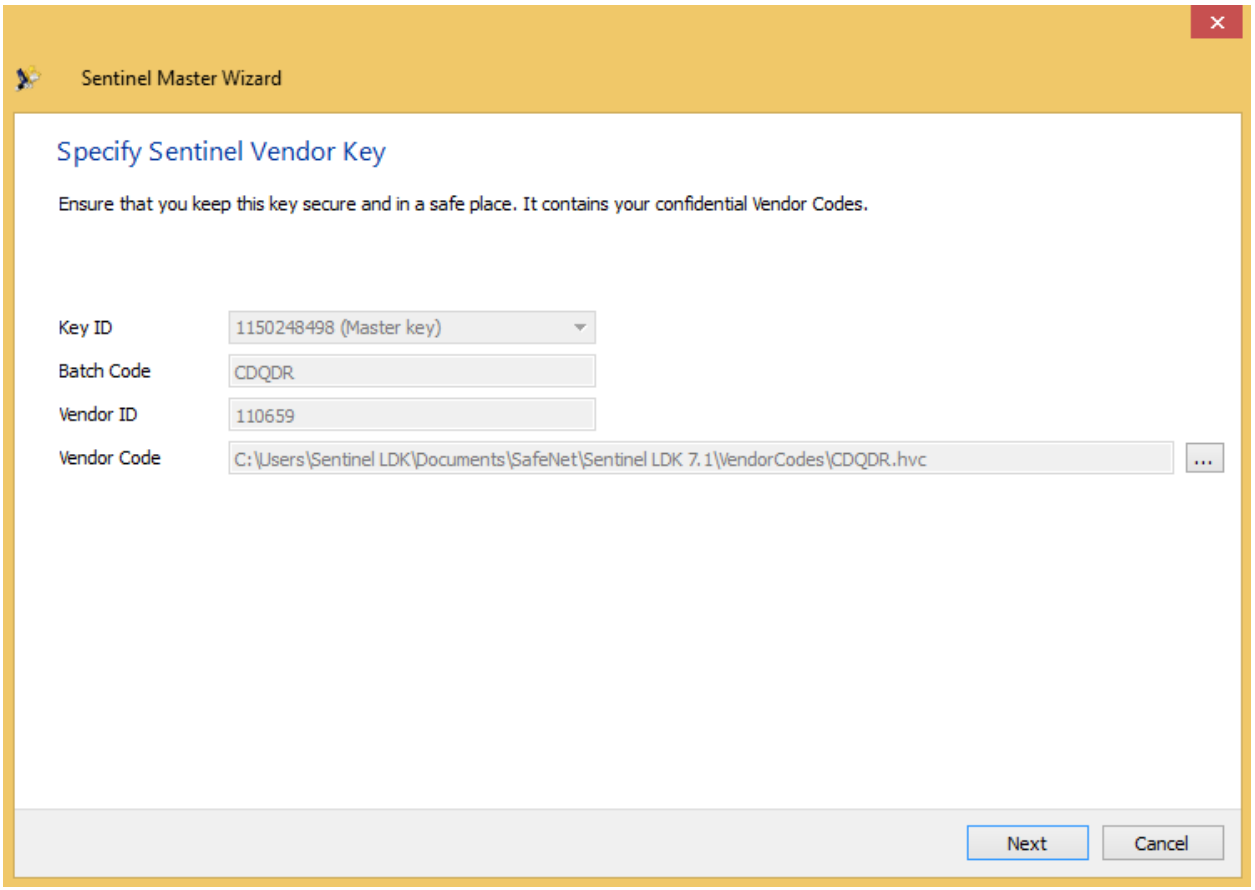
1) Через Интернет – загрузка библиотек и файла с Vendor кодом напрямую с сервера разработчика системы защиты Sentinel LDK.
2) Через MWP файл – процедура выполняется в офлайн режиме при наличии MWP файла.

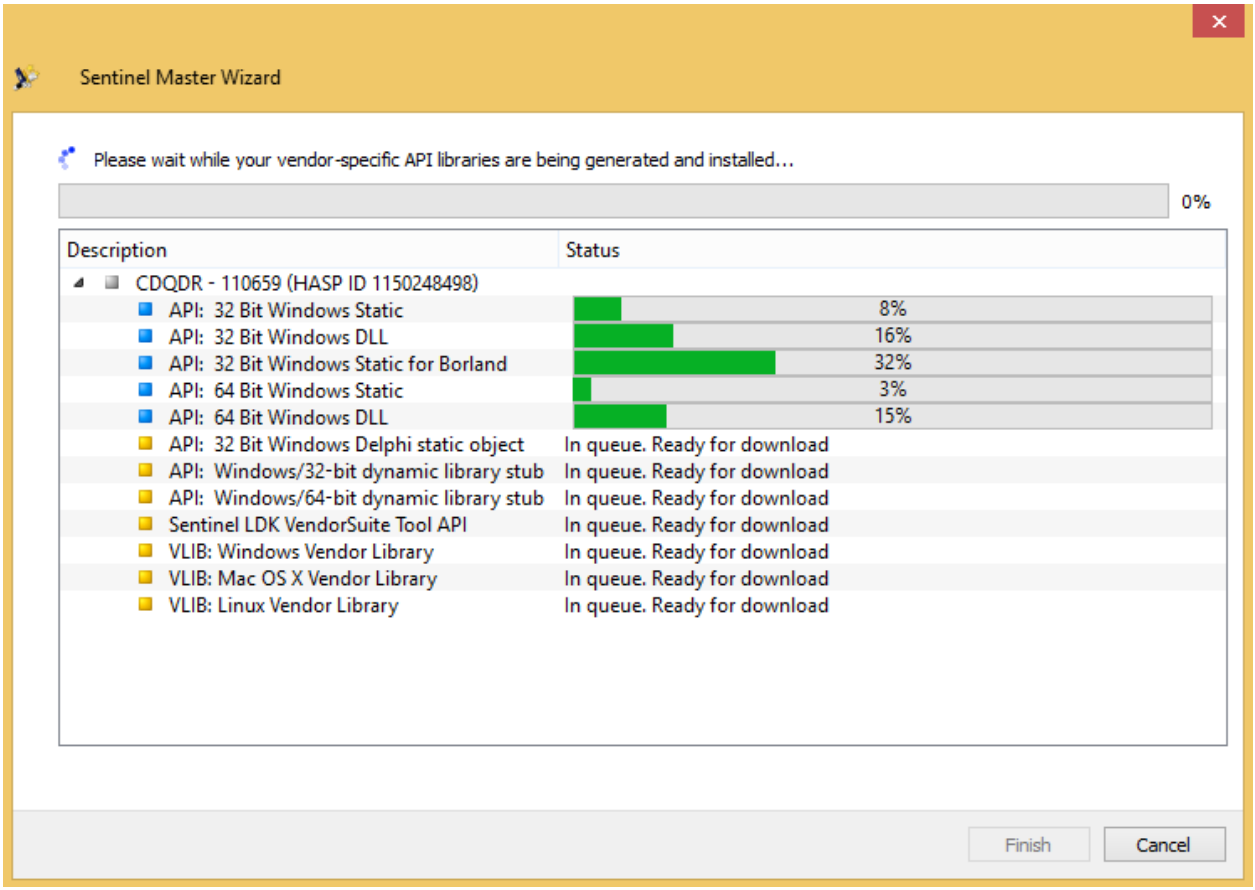


При отсутствии подключения к Интернету появится сообщение с запросом предоставить MWP файл для продолжения процедуры представления служебного ключа:

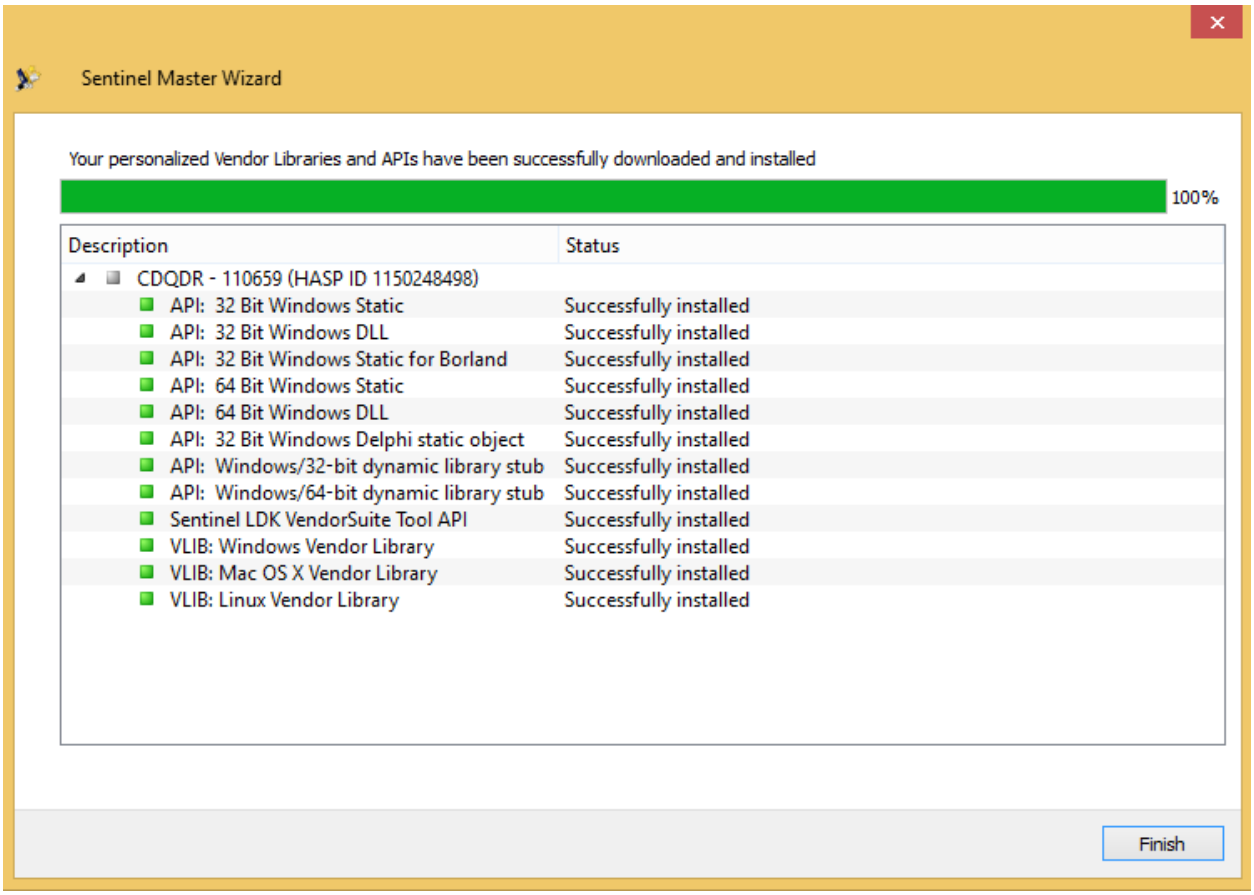


Если процедура осуществляется через MWP файл, то в следующем окне необходимо будет указать путь до MWP файла, после чего в окне с информацией о служебном ключе необходимо нажать кнопку “Next” для старта процедуры:





По окончании процедуры следует нажать кнопку “Finish”.



После этого на ПК станут доступны кастомизированные API библиотеки и файл с Vendor кодом, требуемые для работы с ключами той серии разработчика, для которой проводилась процедура представления служебного ключа.

Кастомизированные API библиотеки и файл с Vendor кодом будут доступны в директории:

*“C:\Users*Учётная_запись_пользователя_под_которой_выполнялась_процедура_представления*\Documents\SafeNet\Sentinel LDK *Версия_используемого_комплекта_разработчика*\”*

Кастомизированные API библиотеки имеют вид:

«hasp_windows_*Vendor_ID *.dll», где «Vendor_ID» – числовой эквивалент Batch кода (наименования серии разработчика).

Файл с Vendor кодом представляет из себя текстовый файл “*Batch_код*.hvc”, к примеру:

*“C:\Users*Учётная_запись_пользователя *\Documents\SafeNet\Sentinel LDK 7.4\VendorCodes\DEMOMA.hvc”* – файл с Vendor кодом для демонстрационной серии разработчика DEMOMA.

3.3. Защита приложений

3.3.1.С помощью Sentinel LDK Envelope

Sentinel LDK Envelope – это утилита, предназначенная для автоматической защиты уже скомпилированного приложения, реализует множество механизмов защиты, таких как:

- Шифрование кода приложения;
- Обфускация кода;

- Механизмы борьбы с отладчиками;
- Привязка защищённого приложения к ключу защиты Sentinel;
- Механизм периодической проверки ключа.

Sentinel LDK Envelope позволяет защищать под Windows:

- Исполняемые файлы Win32 и Win64 приложений;
- JAR и WAR файлы;
- APK файлы*;
- DLL библиотеки, в том числе и .Net сборки,

а также шифровать FLV, SWF и любые другие файлы, для того чтобы с этими файлами можно было работать только из защищённого приложения с ключом.

**Для защиты APK файлов на ПК должен быть также установлен JDK, в настройках Sentinel LDK Envelope должен быть указан путь до его корневой директории.*

Для защиты файлов под операционные системы Linux или Mac OS необходимо воспользоваться Sentinel LDK Envelope под соответствующую платформу.

Sentinel Envelope под Windows существует в двух вариантах:

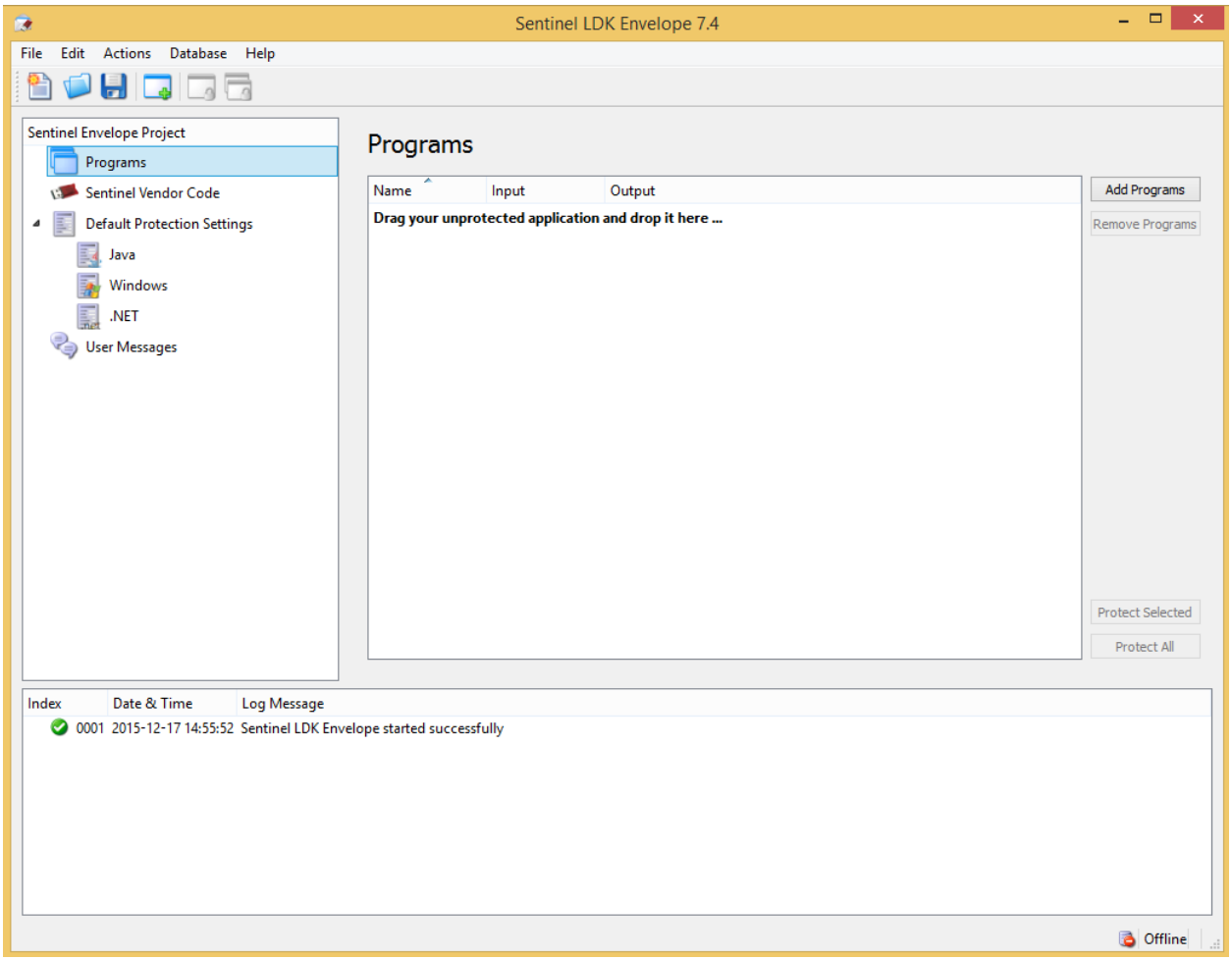
1) GUI Envelope – с графическим интерфейсом;
2) CMD Envelope – с консольным интерфейсом, защита осуществляется с помощью команд. Подробнее об этом можно почитать в документации на английском языке, представленной в комплекте разработчика, файл:

   “C:\Program Files\SafeNet Sentinel\Sentinel LDK\Docs\Manuals & Tutorials\Software Protection and Licensing Guide.pdf”

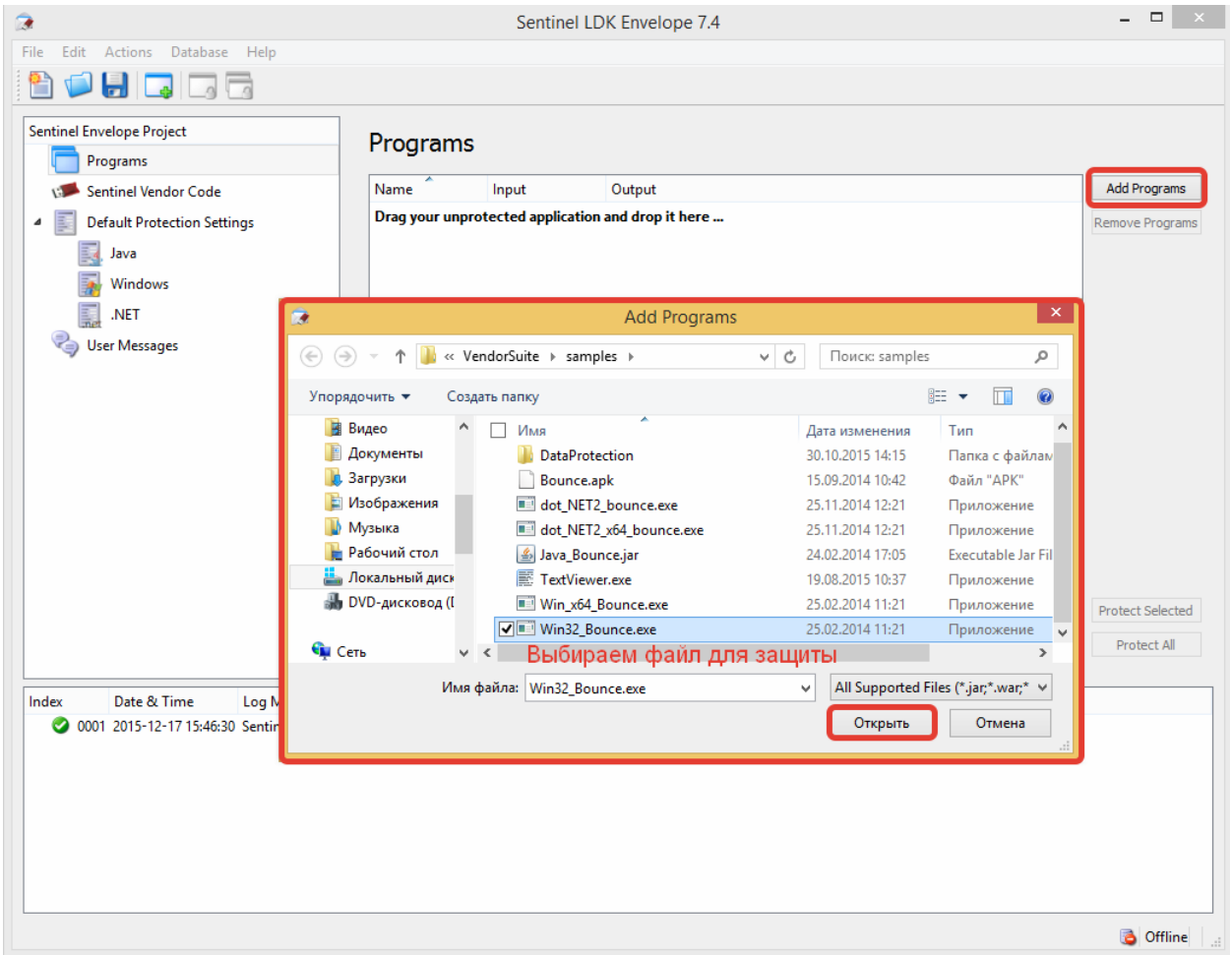
Для защиты приложения с помощью Sentinel LDK Envelope необходимо наличие на ПК подключенного служебного ключа Мастер или Developer.

Процедура защиты приложения выглядит следующим образом:

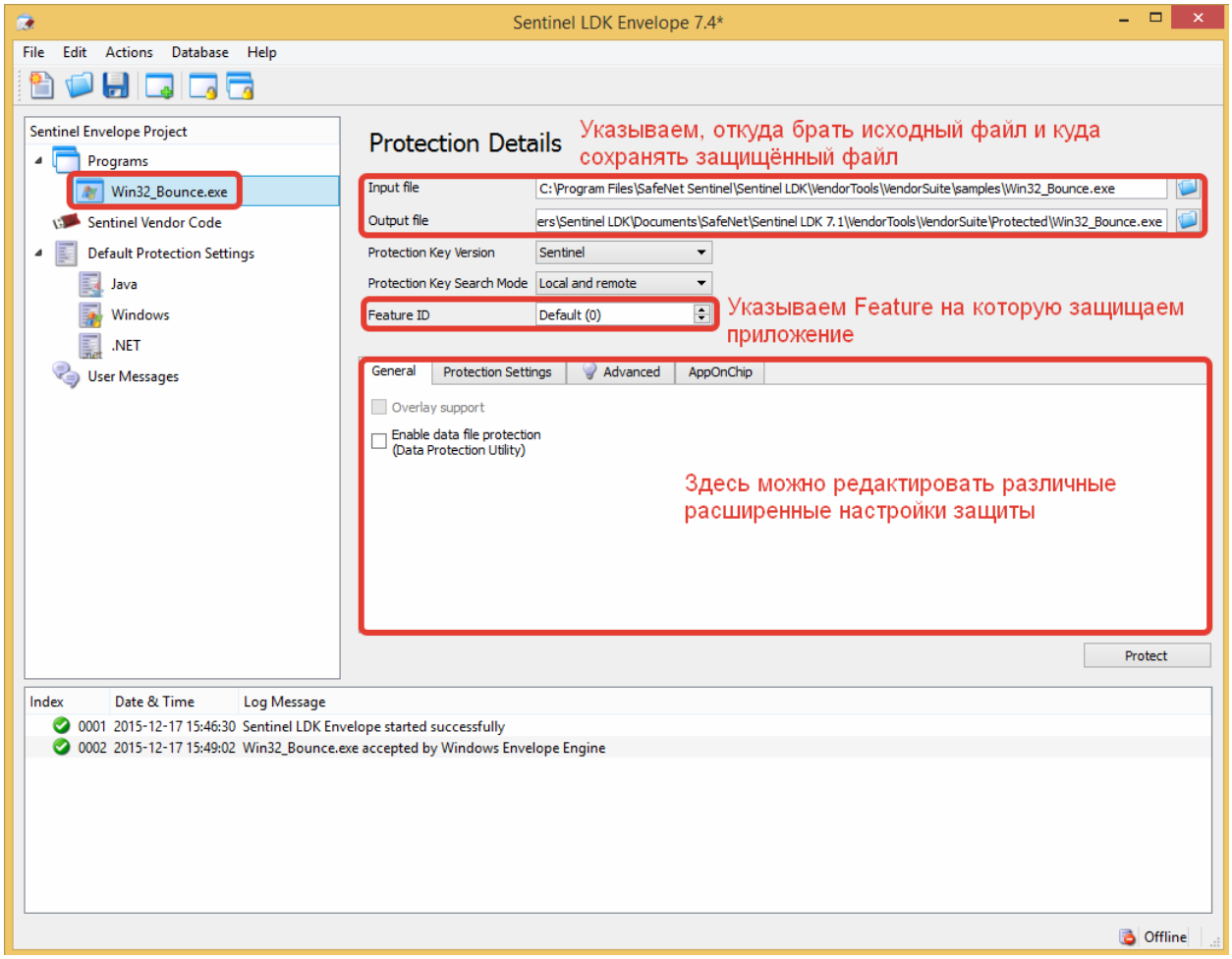
1) Подключаем к ПК служебный ключ.
2) Запускаем утилиту Sentinel LDK Envelope:



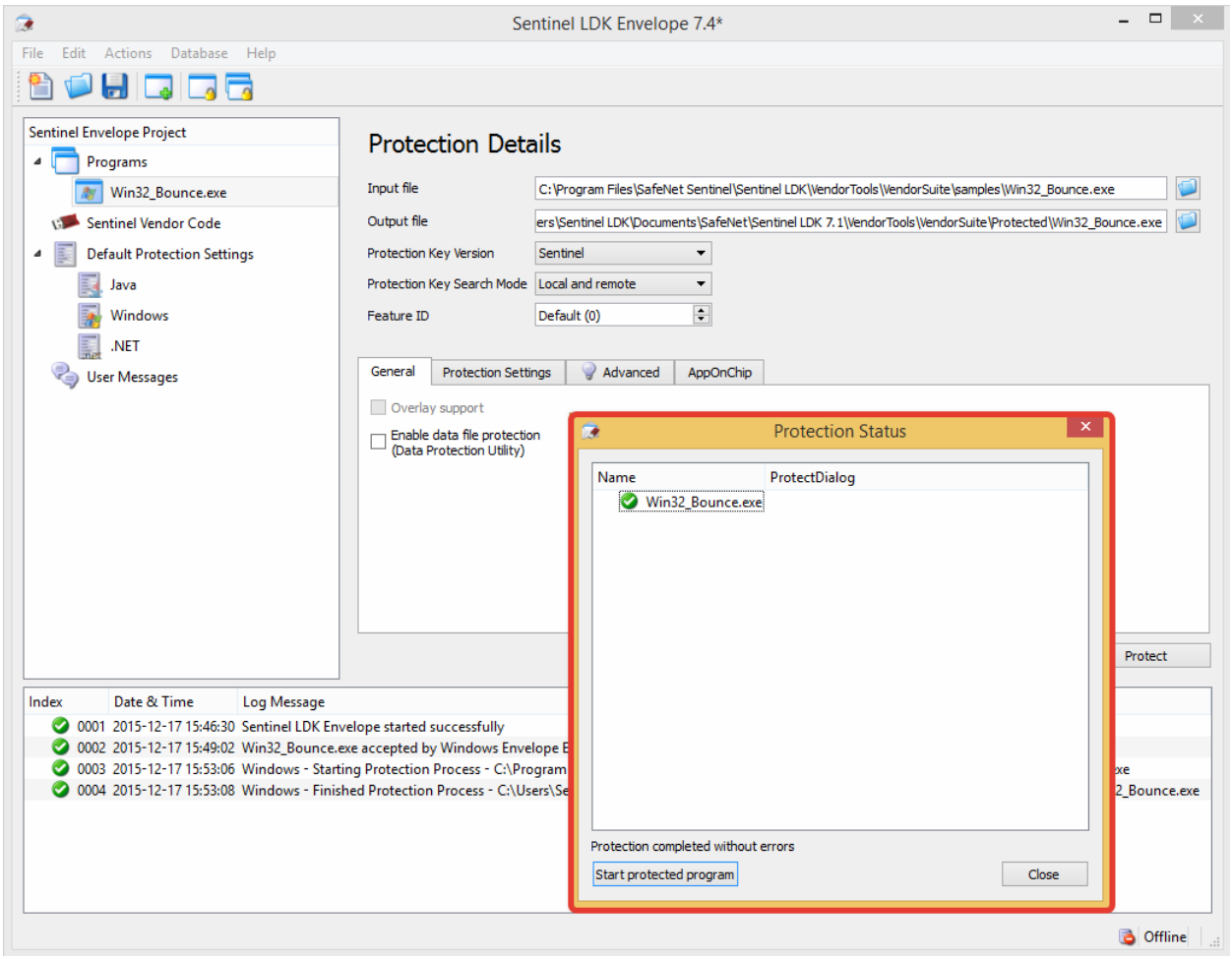
3) Добавляем в него защищаемое приложение:



4) Выставляем требуемые настройки защиты, основные из которых:
   - Пути от исходного файла и до защищённого файла;
   - Feature ID – feature, на которую необходимо защитить приложение (если указать feature, отличную от 0, то в дальнейшем потребуется записать ее в ключ, чтобы защищённое ПО смогло работать с данным ключом защиты);
   - Также на вкладках «Общие», «Настройки защиты», «Дополнительно» и «AppOnChip» (доступно только для Win32 приложений) можно указать дополнительные настройки защиты, подробнее о которых можно почитать в справке к утилите Sentinel LDK Envelope.



5) После этого нажимаем на кнопку «Защитить». Появляется окно со статусом защиты. После успешной защиты становится доступна кнопка «Запустить защищённую программу», нажатие на которую запускает защищённое приложение. Для работы приложения необходимо подключить пользовательский ключ защиты Sentinel с записанной в него feature id, на которую производилась защита приложения.



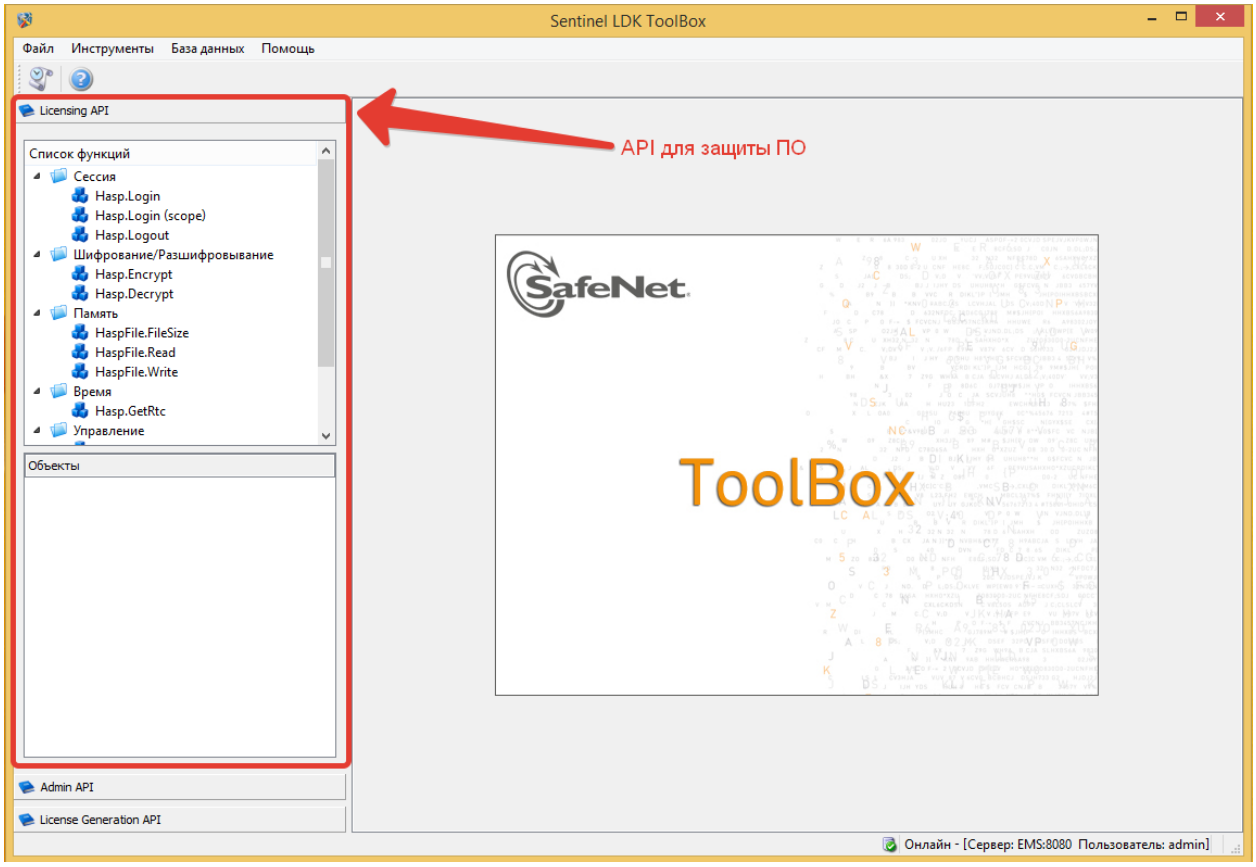
Также защищённое приложение можно найти в директории, куда сохранялся «Целевой файл».

Помимо защищённого файла в директории с защищённым приложением могут оказаться дополнительные файлы (библиотеки и исполняемые файлы), требуемые для работы защищённого приложения, например:

- “hasp_rt.exe” – портативный менеджер лицензий, необходим для работы защищённого приложения с ключом защиты, в том числе по сети, на машине без драйвера. Он должен находиться в той же директории, что и защищённый файл, а также в той директории, где лежит файл, вызывающий этот защищённый модуль.
- “haspvlib_*Vendor_ID*.dll” – кастомизированная API библиотека, необходимая для работы защищённого приложения с программными ключами защиты Sentinel SL. Должна лежать в директории с портативным менеджером лицензий “hasp_rt.exe”.
- “haspdnert.dll” – библиотека, требуемая для работы защищённого с помощью Sentinel LDK Envelope .NET приложения. Должна лежать в той же директории, что и защищённый файл.
- “hasp_windows_demo.dll” – библиотека, требуемая для работы защищённого с помощью Sentinel LDK Envelope .NET или Java приложения. Должна лежать в той же директории, что и защищённый файл.
- “HASPJava.dll” – библиотека, требуемая для работы защищённого с помощью Sentinel LDK Envelope Java приложения. Должна лежать в той же директории, что и защищённый файл.

3.3.2.С помощью Licensing API

Защита через Licensing API строится с помощью 13-ти API функций, каждая из которых представлена в Sentinel LDK Toolbox.



Список функций с описанием:

1) hasp_login. Функция используется для создания сессии с ключом.
   В качестве параметров на вход функция принимает:
   - Feature ID, с которой требуется создать сессию;
   - Vendor Code.

Для работы с ключом защиты требуется обязательное наличие открытой сессии (кроме работы функции hasp_get_info).

2) hasp_login_scope. Функция используется для создания сессии с ключом и даёт возможность задать логику выбора ключа, к которому будет осуществляться подключение.
   В качестве параметров на вход функция принимает:
   - Feature ID, с которой требуется создать сессию;
   - Vendor Code;
   - scope – XML-структуру с параметрами фильтра.

3) hasp_logout . Функция используется для завершения сессии, созданной ранее с ключом.
   В качестве параметров на вход функция принимает:
   - handle – заголовок сессии, получаемый при создании сессии через функции hasp_login или hasp_login_scope.

4) hasp_encrypt . Функция используется для шифрования данных через криптопроцессор ключа.
   В качестве параметров на вход функция принимает:
   - handle – заголовок сессии, получаемый при создании сессии через функции hasp_login или hasp_login_scope;
   - len – размер шифруемого блока данных в байтах (минимум 16 байт, максимум 1024);
   - data – указатель на адрес блока данных в буфере, который необходимо зашифровать.

5) hasp_decrypt . Функция используется для расшифрования данных через криптопроцессор ключа.
   В качестве параметров на вход функция принимает:
   - handle – заголовок сессии, получаемый при создании сессии через функции hasp_login или hasp_login_scope;
   - len – размер расшифровываемого блока данных в байтах (минимум 16 байт, максимум 1024);
   - data – указатель на адрес блока данных в буфере, который необходимо расшифровать.

   Расшифровывать данные необходимо в рамках открытой сессии с той же feature id, на которой данные и зашифровывались, так как закрытый ключ шифрования для каждой feature id уникальный.

6) hasp_get_size. Функция используется для получения размера областей памяти на ключе защиты (в байтах).
   В качестве параметров на вход функция принимает:
   - handle – заголовок сессии, получаемый при создании сессии через функции hasp_login или hasp_login_scope;
   - идентификатор типа памяти, для которой нужно узнать размер.

   Существуют два типа памяти:
   a. Read-Only память (HASP_FILEID_RO) – в эту область памяти можно записывать данные только с использованием мастер-ключа, а читать данные можно через Licensing API. Read-Only память может использоваться для хранения данных, которые не изменяются во время работы приложения.
   b. Read/Write память (HASP_FILEID_RW) – в эту область памяти данные можно записывать и читать как с помощью Licensing API, так и с помощью мастер-ключа. Read/Write память может использоваться для хранения каких-либо динамических данных, используемых в защищаемом ПО.

7) hasp_read. Функция используется для чтения данных из памяти ключа.
   В качестве параметров на вход функция принимает:
   - handle – заголовок сессии, получаемый при создании сессии через функции hasp_login или hasp_login_scope;
   - идентификатор типа памяти;
   - offset – сдвиг, с какого байта памяти следует осуществлять чтение;
   - len – размер читаемого блока памяти в байтах.

8) hasp_write. Функция используется для записи данных в Read/Write область памяти ключа.
   В качестве параметров на вход функция принимает:
   - handle – заголовок сессии, получаемый при создании сессии через функции hasp_login или hasp_login_scope;
   - идентификатор типа памяти (через Licensing API писать можно только в Read/Write область памяти);
   - offset – сдвиг, с какого байта памяти следует осуществлять запись;
   - len – размер записываемого блока памяти в байтах.

9) hasp_get_rtc. Функция используется для получения значения часов реального времени в ключах Sentinel HL Time, Sentinel HL Net Time и программных ключах Sentinel SL.
   В качестве параметров на вход функция принимает:
   - handle – заголовок сессии, получаемый при создании сессии через функции hasp_login или hasp_login_scope.

   Возвращает значение, равное числу секунд, прошедших с 01 января 1970 г. 0:00 часов UTC.

10) hasp_get_info. Функция используется для получения информации о ключах защиты, установленных как на локальном ПК, так и на других ПК в сети, а также для получения информации о менеджерах лицензий.
    В качестве параметров на вход функция принимает:
    - vendor-код;
    - scope – настраиваемые параметры поиска;
    - format – настраиваемый формат вывода получаемой информации.

    Можно использовать для получения C2V-файла с ключа или для получения fingerprint (слепка) системы.

11) hasp_get_sessioninfo. Функция используется для получения информации о ключе в рамках установленной сессии или информации о менеджере лицензий.
    В качестве параметров на вход функция принимает:
    - handle – заголовок сессии, получаемый при создании сессии через функции hasp_login или hasp_login_scope;
    - vendor-код;
    - format – настраиваемый формат вывода получаемой информации.

    Можно использовать для получения C2V-файла с ключа для удалённого обновления или для реализации фоновой проверки на наличие ключа**.**

12) hasp_update . Функция используется для применения V2C-файла с обновлением. Обновление применяется либо к уже существующему ключу, если речь идёт об обновлении лицензий в ключе, либо к системе, если речь идёт об активации нового программного ключа.
    В качестве параметров на вход функция принимает:
    - V2C-массив данных.

    В результате возвращается C2V-массив с состоянием ключа после применения обновления.

13) hasp_transfer . Функция используется для переноса программного ключа или части лицензий из ключа на другой ПК (функционал Rehost и Detach). С её помощью можно выполнять следующие действия:

a. Выполнять полный перенос программного ключа на другой ПК (функционал Rehost);
b. Выполнять перенос части лицензий из программного ключа на другой ПК на время (функционал Detach);
c. Выполнять возврат части лицензий, перенесённых из программного ключа на другой ПК (функционал Cancel Detach).

В качестве параметров на вход функция принимает:

- action – тип выполняемой операции (Rehost, Detach или Cancel Detach);
- scope – XML-структура с настраиваемыми параметрами поиска;
- vendor-код;
- recipient – информация о ПК-получателе лицензии (ПК, куда переносится лицензия) в формате XML-файла с расширением "*.id". Файл "*.id" можно получить, используя либо функцию hasp_get_info, либо функцию hasp_get_sessioninfo. Для Cancel Detach параметр recipient должен принимать значение Null.

Licensing API использует кастомизированные API библиотеки, получаемые для каждого кода разработчика, путём проведения процедуры представления служебного ключа.

Библиотеки можно найти в директории с документами той учётной записи, от которой запускалась утилита Sentinel Master Wizard для проведения процедуры представления служебного ключа, пример директории:

*"C:\Users*Имя_учётной_записи*\Documents\SafeNet\Sentinel LDK *версия_используемого_LDK*\API \Runtime\"*

Также в комплекте разработчика есть примеры работы с различными API, в частности, примеры по работе с Licensing API можно найти в директории:

"C:\Program Files\SafeNet Sentinel\Sentinel LDK\Samples\Runtime\"

Или "Program Files (x86)" если операционная система x64:

"C:\Program Files (x86)\SafeNet Sentinel\Sentinel LDK\Samples\Runtime\"

Все примеры собраны под демонстрационный код разработчика DEMOMA. Чтобы пересобрать пример под свой код разработчика, потребуется заменить кастомизированные API библиотеки и Vendor код на API библиотеки и Vendor код своей серии ключей.

При построении защиты с помощью API реализация механизмов защиты полностью ложится на разработчика приложения.

Некоторые рекомендации для построения более надёжной защиты:

1) Код защиты не должен быть расположен компактно. Он должен быть фрагментирован на отдельные атомарные операции, каковые надлежит рассредоточить по всему защищаемому коду.
2) Реакция защиты должна быть отложенной, а не мгновенной, это затруднит локализацию кода защиты.
3) Vendor код необходимо хранить в зашифрованном виде, расшифровывать непосредственно перед использованием, после использования сразу либо зашифровать

вновь, либо уничтожить расшифрованную копию. Это так же затруднит локализацию кода защиты.

4) Помимо явных точек принятия решения (например, проверка кода возврата и, если ключа нет, выдача MessageBox'a) обязательно должны быть и неявные, т.е. код защиты должен поставлять некие данные, необходимые для работы защищаемого кода. Например, какие-либо значения, нужные для корректной работы программы, можно хранить в зашифрованном виде и по мере необходимости расшифровывать их через ключ, не выполняя явной проверки результата этой операции. Если в код защиты вмешались, результат расшифровывания будет некорректным, и программа будет работать неправильно.
5) Поток, из которого будет осуществляться периодический фоновый опрос ключа, должен быть защищен, или должен выполнять некоторую работу, необходимую для корректного функционирования приложения, иначе можно будет просто предотвратить его запуск, отключив таким образом фоновый опрос ключа.
6) Крайне желательно реализовать второй эшелон защиты, который сразу не будет работать, а будет ждать некоторое время (например, 1 месяц после первого запуска приложения). Когда придет время, второй эшелон активируется и начнет периодически проверять состояние первого эшелона, где реализована вся работа с ключом. В качестве основы второго эшелона можно использовать либо механизм расчета и проверки контрольных сумм, либо применить криптографические hash-функции, либо еще что-нибудь на усмотрение разработчика приложения, кроме работы с ключом, т.к. разные эшелоны защиты должны использовать различные базовые принципы работы.

Помимо описанных рекомендаций можно в код своего приложения встроить API функции для работы с ключом (чтение/запись памяти ключа, шифрование/расшифровывание с использованием криптопроцессора ключа), и затем уже скомпилированное приложение обработать утилитой Sentinel LDK Envelope. Таким образом, в автоматическом режиме для защищаемого приложения будут реализованы все механизмы защиты, доступные при использовании Sentinel LDK Envelope, и помимо этого приложение будет ещё дополнительно работать с ключом защиты через Licensing API. Такой подход позволяет реализовать очень высокий уровень защиты.

Также следует отметить, что трафик между ключом и API библиотеками динамически шифруется с помощью технологии white-box криптографии.

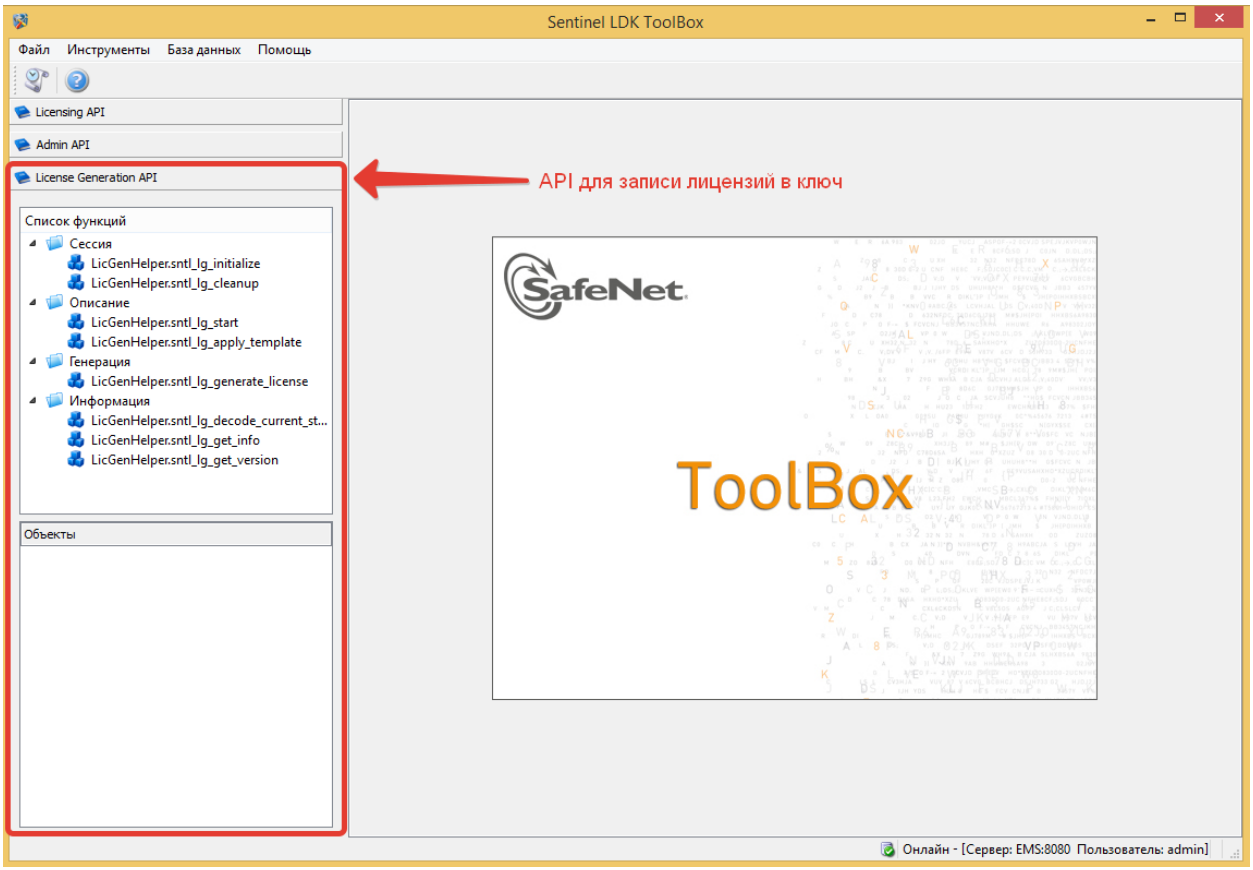
Более подробная информация об API и коды ошибок описаны в документации по API на английском языке. Найти данную документацию на ПК с уже установленным комплектом разработчика можно в файле:

“C:\Program Files\SafeNet Sentinel\Sentinel LDK\API\Runtime\licensing_api_en-US.chm”

3.4. Запись лицензий в ключи защиты с использованием License Generation API

В комплекте разработчика Sentinel LDK есть License Generation API, включающее в себя 8 API функций и позволяющее реализовать свой собственный инструмент для записи лицензий в пользовательские ключи защиты.

Данное API представлено в утилите Sentinel LDK Toolbox:



Для работы с License Generation API необходимо обязательное наличие на ПК подключенного Master ключа той же серии разработчика, что и пользовательский ключ защиты, в который требуется записать лицензии.

Список функций с описанием:

1) sntl_lg_initialize. Функция используется для инициализации библиотеки, необходимой для работы с Master ключом, и возвращает дескриптор, который используется для последующих функций. Данная функция должна выполняться в первую очередь, и только затем могут выполняться остальные функции Licgen API. В результате выполнения данной функции создаётся заголовок сессии (handle), в рамках которой будут выполняться остальные функции.

2) sntl_lg_cleanup. Функция используется для завершения ранее созданной сессии.

   В качестве параметра на вход функция принимает:
   - handle – заголовок сессии.

3) sntl_lg_start. Функция используется для запуска процесса определения лицензии.
   Функция предназначена для проверки корректности лицензионных ограничений, которые будут записаны в ключ.
   В качестве параметров на вход функция принимает:
   - handle – заголовок сессии;
   - start_param – параметр, отвечающий за то, что данная функция резервируется для использования в дальнейшем (по умолчанию следует выставить значение NULL);
   - vendor код;
   - license_type – тип лицензионного ограничения (например, SNTL_LG_LICENSE_TYPE_UPDATE – создает новую лицензию или обновляет существующую лицензию в ключе);

- definition – XML структура лицензионных ограничений, которые будут записаны в ключ;
- current_state – C2V-файл с текущим состоянием ключа.

4) sntl_lg_apply_template. Функция используется для применения шаблона лицензионного ограничения в рамках текущей сессии. Эта функция вызывается после функции sntl_lg_start. Функция может вызываться несколько раз для внесения всех необходимых изменений в состояние лицензии. Например: можно определить продукт с лицензионными ограничениями, используя функцию sntl_lg_start, а затем вызвать sntl_lg_apply_template столько раз, сколько необходимо, чтобы добавить необходимые лицензионные ограничения, другие продукты, или дописать что-либо в память ключа защиты.

   Эта функция проверяет корректность указанных лицензионных ограничений для текущего состояния лицензии в рамках текущей сессии.

   В качестве параметров на вход функция принимает:
   - handle – заголовок сессии;
   - definition – XML-структура, содержащая лицензионные ограничения, которые будут записаны в ключ.

5) sntl_lg_generate_license. Функция используется непосредственно для генерации лицензии. В результате выполнения функция возвращает лицензию в виде XML структуры V2C-файла, которую затем можно применить к ключу посредством функции hasp_update из Licensing API, либо посредством функции sntl_admin_set из Admin API.
   В качестве параметров на вход функция принимает:
   - handle – заголовок сессии;
   - generation_param – параметр, отвечающий за то, что данная функция резервируется для использования в дальнейшем (по умолчанию следует выставить значение NULL);
   - license – указатель на зарезервированную в памяти с помощью функций sntl_lg_start или sntl_lg_apply_template лицензию.

6) sntl_lg_decode_current_state. Функция используется для извлечения из C2V-файла текущего состояния ключа (информацию о текущих лицензиях в ключе, а также об аппаратных характеристиках компьютера в случае, если речь идёт о программных ключах защиты).
   В качестве параметров на вход принимает:
   - handle – заголовок сессии;
   - vendor-код;
   - current_state – XML-структура C2V-файла с текущим состоянием ключа.

7) sntl_lg_get_info. Функция возвращает список моделей ключей, в которые может быть записана данная лицензия. Функция также может вернуть последнее сообщение об ошибке, выданной последней выполненной функцией.
   В качестве параметров на вход функция принимает:
   - handle – заголовок сессии;
   - info_type – тип запрашиваемой информации:
     - SNTL_LG_INFO_CAPABLE_DEVICES – список совместимых с лицензией ключей;
     - SNTL_LG_INFO_CAPABLE_DEVICES – последнее сообщение об ошибке.

8) sntl_lg_get_version. Функция используется для получения информации о версии используемой Licgen API библиотеки.

Более подробная информация об API и коды ошибок описаны в документации по API на английском языке. Найти данную документацию на ПК с уже установленным комплектом разработчика можно в файле:

“C:\Program Files\SafeNet Sentinel\Sentinel LDK\API\Licgen\license_generation_api_en-US.chm”

Примеры работы с API можно найти в директории:

“C:\Program Files\SafeNet Sentinel\Sentinel LDK\Samples\Licgen\”

3.5. Запись лицензий в ключи защиты с использованием Sentinel LDK EMS

Ознакомиться с принципами работы с сервисом Sentinel LDK EMS Вы можете с помощью видео уроков, представленных на нашем сайте:

http://www.safenet-sentinel.ru/demo-zone/

4. Официально поддерживаемые платформы

- **Поддерживаемые платформы для конечных пользователей.**

Sentinel LDK Run-time Environment, защищённое приложение.

Операционная система	Версия Run-time Environment
Windows	Версия 7.41 Sentinel LDK Run-time Environment данная версия имеет “Совместимость с Windows 10” (x86 и x64).
Mac	Версия 7.40
Linux	Версия 7.40

Для поддержки всех новых функциональных возможностей, появившихся в Sentinel LDK 7.4, у конечных пользователей должен быть установлен драйвер версии 7.41 и выше.

Список официально поддерживаемых платформ для драйвера Sentinel LDK Run-time Environment и защищённых с помощью Sentinel LDK 7.4 приложений:

Платформа	Поддерживаемая версия
Windows	• Windows (x86) XP SP3, Windows (x64) XP SP2, Windows Vista SP2,Windows 7 SP1, Windows 8.1, Windows 10, Windows Server 2003 SP2, Windows 2008 SP2, Windows 2008 R2 SP1, Windows Server 2012 R2 Должны быть установлены последние версии Service Pack'ов и все обновления, касающиеся безопасности системы. • (x86 only) Windows XP Embedded standard • (x86 only) Windows 7 SP1 Embedded standard
Mac	• Mac OS X 10.6.8 (32- bit and 64-bit) – не тестировалось в LDK 7.4 • Mac OS X 10.7.5 • Mac OS X 10.8.5 • Mac OS X 10.9.5 • Mac OS X 10.10 • Mac OS X 10.11
Linux	• OpenSUSE 12.3, 13.2 (x86 and x86_64) • Red Hat EL 5.10, 6.5 (x86 and x86_64) – не тестировалось в LDK 7.4 • Red Hat EL 6.6, 7.1 (x86 and x86_64) • Ubuntu Server 12.04.3, 14.04 (x86 and x86_64) • Ubuntu Desktop 12.04.3 (x86 and x86_64) • Debian 6.0.10, 8.1 (x86 and x86_64) • CentOS 6.x (x86 and x86_64) – не тестировалось в LDK 7.4 • CentOS 7.1 (x86 and x86_64) Должны быть установлены последние версии Service Pack'ов и все обновления, касающиеся безопасности системы.
Среда виртуализации	• Virtual Box 4.3.28 • Parallel Desktop 9 for Mac • VMware Player 6.0.3 • Hyper-V Server 2012 R2 (SL only) • VMware Workstation 11.1 • VMware ESXi 5.5 • XEN 4.5 • KVM (RHEL 7.0, Ubuntu 14.04 server, Debian 8.x)
Wine	Драйвер Sentinel LDK Run-time Environment официально поддерживает работу на Linux платформах в Wine 1.7.28.
Linux ARM	Sentinel LDK Embedded поддерживает платформу Linux ARM. Подробнее узнать о поддерживаемых архитектурах и загрузить бесплатную пробную версию вы можете тут: http://www.safenet-inc.com/software-monetization/sentinel-embedded-solutions/

Защита (шифрование файлов) с помощью плагина для Internet Explorer:

- Поддерживаемые версии IE: 8, 9, 10, 11.

**Плагин не устанавливается и не работает на виртуальных машинах.*

Официально поддерживаемые браузеры для работы с Sentinel Admin Control Center:

- Microsoft Internet Explorer (32- bit) versions 8, 9, 10, 11
- Microsoft Edge
- Mozilla Firefox (32- bit) version 22
- Google Chrome (32- bit) version 23 or later
- (Mac) Safari 5.0, 6.0

- **Поддерживаемые платформы для разработчиков.**

Sentinel EMS Service.

Платформа	Поддерживаемая версия
Windows	Windows (x86) XP SP3, Windows (x64) XP SP2, Windows Vista SP2,Windows 7 SP1, Windows 8.1, Windows 10, Windows Server 2003 SP2, Windows 2008 SP2, Windows 2008 R2 SP1, Windows Server 2012 R2

Sentinel EMS Database.

Платформа	Поддерживаемая версия
Windows	• Microsoft SQL Server 2005 x86/x64 • Microsoft, SQL Server 2005 Express Edition (must be enabled for remote connections) x86/x64 • Microsoft SQL Enterprise 2008 x86/x64 • Microsoft SQL Enterprise 2008 R2 x86/x64 • Microsoft SQL Server 2012 x86/x64 • Microsoft SQL Server 2012 R2 x86/x64 • Microsoft SQL Server 2014 x86/x64 Microsoft SQL Server 2008 R2 Express Edition может быть установлен вместе с Sentinel EMS. Инсталлятор данной версии Microsoft SQL Server доступен в дистрибутиве комплекта разработчика Sentinel LDK.

Официально поддерживаемые браузеры для работы с Sentinel EMS:

- Microsoft Internet Explorer versions 8, 9, 10, 11
- Mozilla Firefox (32- bit) version 40 или более поздняя

**Все действия с ключами, требующие запуска Java плагина (такие, как прошивка ключей, операция Check in C2V, Recycle или онлайн активация ключа) необходимо выполнять в 32-битной версии браузера. Все остальные действия можно выполнять как в 32-битной, так и в 64-битной версии браузера.*

Sentinel LDK Vendor Tools.

Платформа	Поддерживаемая версия
Windows	Windows (x86) XP SP3, Windows (x64) XP SP2, Windows Vista SP2,Windows 7 SP1, Windows 8.1, Windows 10, Windows Server 2003 SP2, Windows 2008 SP2, Windows 2008 R2 SP1, Windows Server 2012 R2 Поддерживаемое разрешение экрана 1280x1024 пикселей, 24 bit глубина цвета. **Sentinel LDK Envelope: для защиты и исполнения .NET приложений под Windows 8, 8.1 или Windows Server 2012 R2, необходимо установить Microsoft .NET Framework 3.5.* ***Приложения, защищаемые с использованием технологии AppOnChip, не будут работать в среде виртуализации Virtual Box версии ниже чем 4.3.28.* ****Плагин Data File Protection, используемый для отображения зашифрованных FLV или SWF файлов, генерируется в процессе проведения процедуры представления Master ключа. Если планируется использовать данный плагин, необходимо на ПК, где будет выполняться процедура представления Master ключа, установить Microsoft .NET Framework версии 3.5 или выше.*
Mac	• Mac OS X 10.8.5 • Mac OS X 10.9.2 • Mac OS X 10.10 • Mac OS X 10.11 **Поддерживаются приложения собранные в среде Cocoa framework.*
Linux	• OpenSUSE 12.3 (x86 and x86_64) • Red Hat EL 7.1 (x86 and x86_64) • Ubuntu Server 12.04.03, 14.04 (x86 and x86_64) • Ubuntu Desktop and Server 12.04.03 (x86 and x86_64) • Debian 6.0.10, 8.1 (x86 and x86_64) • CentOS 7.1 (x86 and x86_64) Должны быть установлены последние версии Service Pack'ов и все обновления, касающиеся безопасности системы.

Поддерживаемые версии для Windows CE.

Драйвер Sentinel LDK Run-time Environment (версии 5.95) и Sentinel LDK Envelope поддерживают Windows CE версий 5.0 и 6.0.

Примеры работы с API.

Примеры	Поддерживаемые версии платформ
Примеры по работе с Sentinel Licensing API – 4D пример под Mac OS X	• Mac OS X 10.6.8 (32- bit and 64-bit) • Mac OS X 10.7.5 • Mac OS X 10.8.5
Примеры по работе с Sentinel Activation API	• Windows XP • Windows Vista • Windows 7 (32- bit and 64-bit) • Windows Server 2003 • Windows 2008
Примеры по работе с Sentinel Activation API под Mac OS X	• Mac OS X 10.6.8 (32- bit and 64-bit) • Mac OS X 10.7.5 • Mac OS X 10.8.5

Примеры	Поддерживаемые языки и среды разработки	
Примеры по работе с Sentinel Licensing API	**Язык программирования**	**Среды разработки**
	AutoCAD	• AutoCAD 2009 • AutoCAD 2010 • AutoCAD 2014
	C	• Visual Studio 2005 • Visual Studio 2008 • Visual Studio 2010 • Visual Studio 2013* • Visual Studio 2015 • C++ Builder • Developer Studio 2006
	C++	• Visual Studio 2005 • Visual Studio 2008 • Visual Studio 2010 • Visual Studio 2013* • Visual Studio 2015 • C++ Builder • Developer Studio 2006 • gcc Для сборки примеров под x64 используйте VS 2008, убедитесь, что x64 компилятор был установлен вместе с VS.
	C#	• Visual Studio 2013* • Visual Studio 2015
	Delphi	• Delphi 2007 • Developer Studio 2006
	Java	• Java Developer Kit 1.7 • Java Developer Kit 1.8

	Visual Basic .NET	• Visual Studio 2013* • Visual Studio 2015
	4D	4D v11 SQL
Примеры по работе с Sentinel Licensing API – под Mac OS X	Java	Java Developer Kit 1.6
	4D	4D v11 SQL
	C	gcc
Примеры по работе с Sentinel Licensing API – под Linux	Java	Java Developer Kit 1.6
	C	gcc
	C++	gcc
Примеры по работе с Sentinel Licensing API – под Android	Java	Java Developer Kit 1.8
Примеры по работе с Sentinel License Generation API	C, C#, Visual Basic .NET	Microsoft Visual Studio 2013*, 2015
	Java	Java Developer Kit 1.8
Примеры по работе с Sentinel LDK Run-time Environment Installer API	C	• Visual Studio 2008 • Visual Studio 2005 • Visual Studio 2010 • Visual Studio 2012
	MSI	• Wise Installer 7.1 • InstallShield 2012 Spring или более новая версия *Внимание: Предоставляемые решения могут использоваться только с InstallShield 2013 Spring или более свежей версии.*
Примеры по работе с Sentinel Activation API под Windows	C	• Visual Studio 2003 • Visual Studio 2005 • Visual Studio 2008 • Visual Studio 2010 • Visual Studio 2012 Возможно, потребуется конвертировать пример под используемую версию VS.
	Java	Java Developer Kit 1.6
Примеры по работе с Sentinel Activation API под Mac OS X	Java	Java Developer Kit 1.6

Примеры по работе с Sentinel Activation API под Linux	Java	Java Developer Kit 1.6
Примеры по работе с Sentinel Admin API	Java	Java Developer Kit 1.8
	C, C++, C#, Visual Basic .NET	Microsoft Visual Studio 2013*, 2015
Envelope Runtime API	C#	Microsoft Visual Studio 2013*, 2015
**Проекты из Microsoft Visual Studio 2015 необходимо импортировать, чтобы использовать их в Microsoft Visual Studio 2013*		

5. Контакты технической поддержки

Телефон: +7 (495) 783-28-78

Email: support-ru@safenet-inc.com

Форум технической поддержки:

http://www.safenet-sentinel.ru/helpdesk/forum/

Центр загрузки:

http://safenet-sentinel.ru/helpdesk/download-space/

http://sentinelcustomer.safenet-inc.com/sentineldownloads/