Описание объектов PKCS #11

В этом разделе приведен перевод стандарта PKCS #11 в части касающейся объектов, поддерживаемых устройствами Рутокен, а также информация об объектах и атрибутах, определенных производителем устройств Рутокен.

Стандарт PKCS #11 различает несколько классов объектов, определяемых типом данных **CK_OBJECT_CLASS**. Объекты содержат набор атрибутов, каждый из которых имеет определенное значение. Каждый атрибут, которым обладает объект, имеет только одно значение. Следующий рисунок отображает высокоуровневую иерархию объектов стандарта и некоторых поддерживаемых ими атрибутов (рисунок 1).

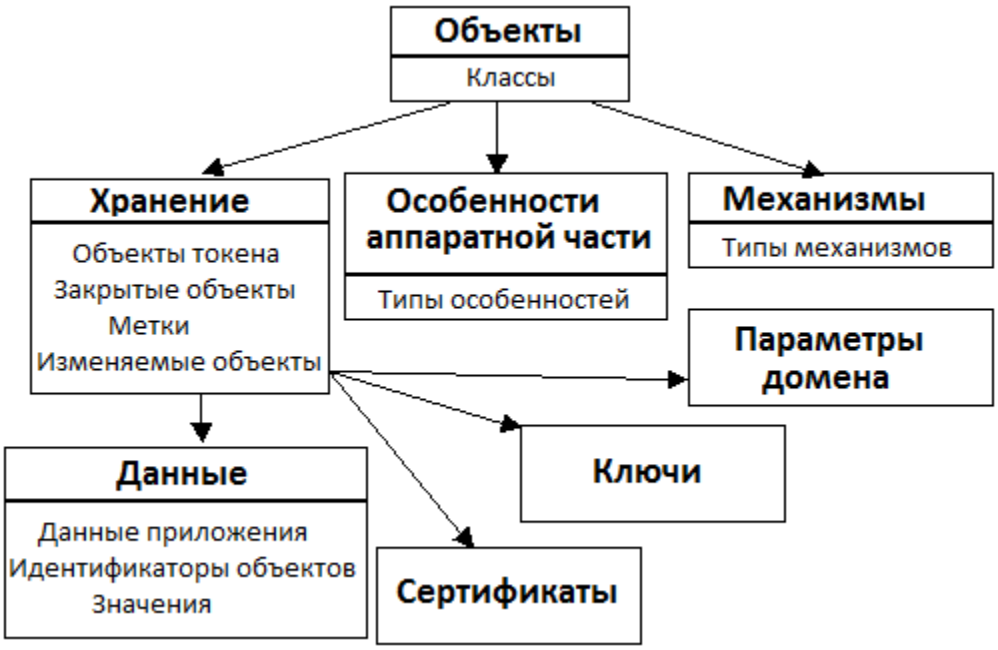

Рисунок 2.1 – Иерархия атрибутов объектов

Стандарт PKCS #11 предоставляет функции для создания, уничтожения и копирования самих объектов, а также для получения и изменения значений их атрибутов. Некоторые из криптографических функций (например, **C_GenerateKey**) также создают ключевые объекты, содержащие результаты их выполнения.

Объекты стандарта всегда четко определены – т.е. объект всегда содержит все требуемые атрибуты, и атрибуты всегда согласуются друг с другом с момента создания объекта. Это является отличием от объектно-ориентированной парадигмы, где объект не имеет других атрибутов, кроме, может быть, класса, которым был создан, и некоторые время продолжает быть неинициализированным. В PKCS #11 объекты всегда инициализируются.

Таблицы на протяжении большей части этого раздела определяют для каждого атрибута тип данных значения атрибута и значение (смысл) атрибута, который может включать начальное значение по умолчанию. Некоторые типы данных определены подробно стандартом (например, **CK_OBJECT_CLASS**). Значения атрибутов могут быть следующих типов:

Byte array	случайная строка (массив) данных типа **CK_BYTE**
Big integer	строка данных типа **CK_BYTE**, состоящая из беззнаковых целых чисел произвольной длины со наиболее значимым первым байтом (например, число 32768 представляется как 2-байтовая строка 0x80 0x00)
Local string	незаполненная строка данных типа **CK_CHAR** без завершающего нуля
RFC2279 string	незаполненная строка данных типа **CK_UTF8CHAR** без завершающего нуля

Токен может содержать несколько идентичных объектов, то есть разрешается иметь одинаковые значения для всех атрибутов двум и более объектам.

В большинстве случаев каждый тип объекта стандарта PKCS#11 содержит полный набор атрибутов. Некоторые из этих атрибутов имеют значения по умолчанию, и не нуждаются в определении при создании объекта; некоторые из значений по умолчанию могут даже содержать пустые строки (“”). Тем не менее, объект содержит эти атрибуты. Объект может иметь только одно значение для каждого атрибута, даже если атрибут определяется производителем и его цель его использования выходит за рамки стандарта.

Кроме атрибутов стандарта PKCS#11, объект также может обладать определяемыми производителем атрибутами, смысл и значения которых не описаны в PKCS#11.

Создание и изменение объектов

Все функции, которые создают, изменяют или копируют объекты, используют в качестве одного из своих аргументов шаблон, который специфицирует значения атрибутов. Криптографические функции, которые создают объекты, также могут сами вносить значения некоторых дополнительных атрибутов; какие именно атрибуты будут задаваться вызовом криптографической функции зависит от того, какой используется механизм. В любом случае все требуемые атрибуты, поддерживаемые классом объектов и не имеющие значений по умолчанию, должны быть указаны при создании объекта в шаблоне или самой функцией.

Создание объектов

Объекты могут быть созданы с помощью функций стандарта PKCS#11 **C_CreateObject**, **C_GenerateKey**, **C_GenerateKeyPair**, **C_UnwrapKey** и **C_DeriveKey**. Кроме того, копирование уже существующих объектов (с помощью функции **C_CopyObject**) также создает новый объект, но этот тип создания объектов на данный момент **не поддерживается** устройствами Рутокен.

Для создания объекта этими функциями необходимо предоставить соответствующий шаблон.

1. Если предоставленный шаблон определяет значение некорректного атрибута, то объект не создается, и возвращается код ошибки CKR_ATTRIBUTE_TYPE_INVALID. Атрибут считается корректным, если они описан стандартом PKCS #11 или если он является дополнительным атрибутом, описанным производителем и поддерживающимся библиотекой и токеном.
2. Если предоставленный шаблон определяет некорректное значение для корректного атрибута, то объект не создается, и возвращается код ошибки CKR_ATTRIBUTE_VALUE_INVALID. Корректные значения атрибутов стандарта PKCS #11 описаны в стандарте PKCS #11.
3. Если предоставленный шаблон определяет значение для атрибута «только для чтения», то объект не создается, и возвращается код ошибки CKR_ATTRIBUTE_READ_ONLY.
4. Если значений атрибута в предоставленном шаблоне вместе со значениями по умолчанию и значениями, задаваемыми функцией при создании объекта, оказывается недостаточно для полного определения объекта, то объект не создается, и возвращается код ошибки CKR_TEMPLATE_INCOMPLETE.
5. Если значения атрибута в поддерживаемом шаблоне вместе со значениями по умолчанию и значениями, задаваемыми функцией при создании объекта, являются несовместимыми, то объект не создается, и возвращается код ошибки CKR_TEMPLATE_INCONSISTENT. Значения атрибутов считаются несовместимыми, если не все из них удовлетворяют требованиям токена, хотя каждое значение по отдельности является корректным согласно стандарту PKCS #11. Первым примером несовместимого шаблона является шаблон, который задает два разных значения для одного и того же атрибута. Другим примером может служить попытка создать объект секретного ключа с атрибутом, который используется при создании различных типов открытых или закрытых ключей, но не секретных ключей. И наконец, последним примером может быть шаблон с атрибутом, который не соответствует требованиям спецификации токена. Заметим, что в последнем примере несовместимый шаблон является зависимым от токена – для другого токена он может *не* быть несовместимым.
6. Если поддерживаемый шаблон определяет одно и то же значение какого-либо атрибута больше одного раза (или шаблон определяет то же самое значение атрибута, которое задает функция, создающая объект), то дальнейшее развитие ситуации стандартом не определено. Попытка создать объект может быть удачной – тогда создается объект так, как если бы значения его атрибутов были бы заданы один раз, или эта попытка может закончиться неудачей, и тогда возвращается код ошибки CKR_TEMPLATE_INCONSISTENT. Разработчики библиотек рекомендуют делать поведение библиотек таким же, как если бы атрибут встретился в шаблоне только раз; разработчики приложений настоятельно рекомендуют никогда не определять какой-либо атрибут в одном и том же шаблоне более одного раза.

Если при попытке создать объект возникает несколько перечисленных ситуаций одновременно, то возвращаемый код ошибки может быть любым из вышеперечисленных.

Изменение объектов

Объекты могут изменяться с помощью функции **C_SetAttributeValue** стандарта PKCS#11. Шаблон, предоставляемый для функции **C_SetAttributeValue**, может содержать новые значения для атрибутов, которыми объект уже обладает; значения для атрибутов, которыми объект еще не обладает; или и те и другие сразу.

Некоторые атрибуты объекта могут изменяться после создания объекта; некоторые, наоборот, не могут. Кроме того, атрибуты, которые определяются стандартом как изменяемые, могут на самом деле для некоторых видов токенов *не* быть изменяемыми. То есть, если атрибут в стандарте PKCS#11 описан как изменяемый, то это в действительно означает, что атрибут является изменяемым в рамках спецификации самого стандарта. Токен может в действительности не поддерживать изменение некоторых атрибутов. Является ли какой-то атрибут объекта на данном токене изменяемым, зависит от значений определенных атрибутов объекта. Например, значение атрибута объекта секретного ключа **CKA_SENSITIVE** может быть изменено с CK_FALSE на CK_TRUE, но не наоборот.

Все сценарии, описанные в части «Создание объектов», включая коды возвращаемых ошибок, применимы для изменения объектов с помощью функции **C_SetAttributeValue**, кроме ситуации с неполным шаблоном.