

S-100 – Часть 15

Схема защиты данных

Содержание

15-1 Назначение	4
15-2 Нормативные ссылки	4
15-3 Общее описание	5
15-4 Участники схемы защиты	6
15-4.1 Администратор схемы	6
15-4.2 Серверы данных	6
15-4.3 Клиенты данных	7
15-4.4 Производители оригинального оборудования	7
15-4.5 Координатор домена	8
15-4.6 Взаимоотношения участников	8
15-5 Сжатие и пакетирование данных	10
15-5.1 Обзор	10
15-5.2 Алгоритм сжатия	10
15-5.3 Кодирование	10
15-6 Кодирование данных	10
15-6.1 Какие данные кодируются?	10
15-6.2 Как они кодируются?	11
15-6.2.1 Алгоритм кодирования	11
15-6.2.2 Педдинг кодирования	11
15-6.2.3 Шифрование AES режим CBC	12
15-6.2.4 Режим CBC AES – вектор инициализации	13
15-6.2.5 Примеры AES	14
15-7 Шифрование и лицензирование данных	15
15-7.1 Введение	15
15-7.2 Преобразование битовых строк в целые числа	16
15-7.2.1 Преобразование битовой строки в целое число	16
15-7.2.2 Преобразование целого числа в битовую строку	17
15-7.2.3 Преобразование беззнакового целого числа в шестнадцатеричное текстовое представление	18
15-7.2.4 Преобразование шестнадцатеричного текста в целое число без знака	19
15-7.3 User Permit	19
15-7.3.1 Определение user permit	20
15-7.3.2 Формат M_KEY	21
15-7.4 Data Permit	22
15-7.4.1 Файл Permit (PERMIT.XML)	22
15-7.4.2 Файл Permit – Содержание заголовка	24
15-7.4.3 Секции продукта и поля записей разрешения	24
15-7.4.4 Определение записи разрешения	25
15-7.4.5 Подписи файла Permit	25
15-7.4.6 Пример файла PERMIT.XML	26
15-8 Аутентификация данных	26
15-8.1 Введение в проверку подлинности и целостности данных.....	27
15-8.2 Настройка схемы защиты данных, регистрация сервера данных и последовательность аутентификации	28
15-8.3 Проверка цифровых подписей	29
15-8.4 Форматы данных и стандарты цифровых подписей, ключей и	

сертификатов	30
15-8.5 Создание ключевых материалов и запросов на подписание сертификатов (подписанные публичные ключи)	33
15-8.5.1 Установка SA	33
15-8.5.2 Настройка сервера данных	33
15-8.6 Пример цифрового сертификата	34
15-8.7 Создание цифровых подписей сервером данных.....	35
15-8.8 Дополнительные цифровые подписи	37
15-8.9 Проверка целостности и цифровой идентичности данных с цифровой подписью S-100	38
15-8.10 Спецификации MRN	39
15-8.11 Метаданные каталога обмена и спецификация одиночного элемента схемы	40
15-8.11.1 S100_SE_CertificateContainerType	41
15-8.11.2 StandaloneDigitalSignature	41
15-8.11.3 S100_SE_DigitalSignature.....	42
15-8.11.4 S100_SE_SignatureOnData	42
15-8.11.5 S100_SE_SignatureOnSignature	42
15-8.11.6 DataStatus	43
15-8.11.7 S100_SE_DigitalSignatureReference	43
15-9 Глоссарий Схемы защиты данных S-100 и компьютерные термины	45

15-1 Назначение

S-100 часть 15, далее называемая как 'Схема защиты данных' или 'Схема защиты', описывает рекомендуемый стандарт защиты гидрографической или пространственной информации на основе универсальной гидрографической модели ИНО S-100. В ней определяются конструкции и операционные процедуры обеспечения безопасности, которые должны соблюдаться для обеспечения надлежащего функционирования системы защиты и предоставления спецификаций, позволяющих участникам создавать соответствующие системы и распространять данные безопасным и коммерчески жизнеспособным образом.

15-2 Нормативные ссылки

Для применения этого документа требуются следующие справочные документы. Что касается датированных ссылок, то применяется только упомянутое издание. Для недатированных ссылок применяется последнее издание справочного документа (включая поправки).

FIPS Publication 81, *DES Modes of Operation*, National Institute of Standards and Technology www.itl.nist.gov/fipspubs/fip81.htm

FIPS Publication 180-4, *Secure Hash Standard (SHS)*
<https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf>

FIPS Publication 186, *Digital Signature Standard (DSS)*
www.itl.nist.gov/div897/pubs/fip186.htm

ISO/IEC 18033-3, *Information technology – Security techniques – Encryption algorithms – Part 3: Block ciphers (AES)*

Open SSL Cryptography and SSL/TLS Toolkit <https://www.openssl.org/>

PKCS#10 v1.7, *Certification Request Syntax Specification*
<https://tools.ietf.org/html/rfc2986>

RFC 1423, *Privacy Enhancements for Internet Electronic Mail: Part III: Algorithms, Modes and Identifiers* <ftp://ftp.isi.edu/in-notes/rfc1423.txt>

RFC 2451, *The ESP CBC-Mode Cipher Algorithms* <https://tools.ietf.org/html/rfc2451>

RFC 2459 version 3, *Internet X.509 Public-key infrastructure and attribute certificate frameworks* <https://tools.ietf.org/html/rfc2459>

RFC 5651, *Cryptographic Message Syntax (CMS)*, ITU International Telecommunication Union <https://tools.ietf.org/html/rfc5652#section-6.3>

RFC 4647, *Base 64 Encoding*. <https://datatracker.ietf.org/doc/html/rfc4648#section-4>

OSI networking and system aspects – Abstract Syntax Notation One (ASN.1), ITU International Telecommunication Union <https://www.itu.int/ITU-T/studygroups/com17/languages/X.680-0207.pdf>

X.509 Version 3, *Information Technology – Open Systems Interconnection – The Directory: Authentication Framework*, International Telecommunication Union

15-3 Общее описание

В этой части описывается метод обеспечения безопасности цифровых навигационных, гидрографических и пространственных продуктов и информации. Защита данных преследует три цели:

1. Защита от пиратства: Предотвращение несанкционированного использования данных путем шифрования информации о продукте.
2. Селективный доступ: Ограничение доступа только к продуктам, на которые клиент получил лицензию.
3. Аутентификация: Обеспечение гарантий того, что продукты были получены из официальных источников.

Защита от пиратства и селективный доступ обеспечиваются путем шифрования продуктов и предоставления разрешений на расшифровку данных. Разрешение на использование данных имеет срок годности, чтобы позволить доступ к продуктам в течение лицензионного периода. Серверы данных шифруют цифровые продукты перед передачей их клиенту данных. Затем зашифрованные продукты дешифруются системой конечных пользователей (например, ECDIS/ECS) для их переформатирования и импорта во внутренний формат системы (например, SENC). Аутентификация осуществляется с помощью цифровых подписей, применяемых к файлам продукта.

Схема безопасности конкретно не определяет, как можно защитить информацию о продукте после того, как она загружена в приложение конечного пользователя. Это уже ответственность производителя оборудования (OEM).

Схема позволяет массовое распространение защищенных наборов данных на жестких носителях (например, DVD) и может быть доступна и использоваться всеми клиентами, имеющими действующие лицензии, содержащие набор разрешений на данные. Селективный доступ к отдельным продуктам поддерживается путем предоставления пользователям набора лицензионных разрешений на данные, содержащие зашифрованные ключи для доступа к наборам данных. Эта лицензия создается с использованием уникального аппаратного идентификатора целевой системы и уникальна для каждого клиента данных. Следовательно, лицензии не могут быть обменены между отдельными клиентами данных.

Схема защиты предназначена для передачи данных в виде файлов между сторонами. Передача потоковых данных может осуществляться различными методами. Потоковые данные описаны в S-100 часть 14. Схема защиты S-100, описанная в этой части, привязана к значению "S100p15" в элементе protectionScheme каталога обмена CATALOG.XML.

Схема использует опционный алгоритм сжатия, чтобы уменьшить размер набора данных. Незашифрованные файлы продуктов содержат много повторяющейся информации; например, координаты. Поэтому сжатие всегда применяется до того, как файл продукта будет зашифрован и распаковывается после соответствующей расшифровки в системе клиента.

15-4 Участники схемы защиты

Существует несколько типов пользователей схемы:

- Администратор схемы (SA), который только один;
- Сервер данных (DS), которых может быть много;
- Клиент данных (DC), которых может быть много;
- Производитель оригинального оборудования (ОЕМ), которых может быть много;
- Координаторы доменов, которых может быть много.

Более подробное объяснение этих терминов приводится ниже. Подробная информация о ролях каждого из участников схемы определяется ИНО, выступающей в качестве администратора схемы.

15-4.1 Администратор схемы

Администратор Схемы (SA) несет исключительную ответственность за поддержание и координацию Схемы защиты. Функции SA выполняет Международная гидрографическая организация (ИНО) от имени государств - членов ИНО и других организаций, участвующих в Схеме защиты. Эти организации могут играть координирующую роль в области морских продуктов, например, IMO и IALA. ИНО как SA устанавливает процедуры с операторами доменов продуктов, использующими систему защиты для защиты своих продуктов. Эти процедуры позволяют координаторам доменов подписывать цифровые сертификаты, используемые их организациями-членами для участия в Схеме защиты.

SA отвечает за контроль членства в Схеме и обеспечение того, чтобы все участники действовали в соответствии с установленными процедурами. SA поддерживает цифровой корневой сертификат верхнего уровня, используемый для управления схемой защиты и образующий корневую идентификацию в цепочке аутентификации.

SA отвечает за распространение идентификатора производителя (M_ID) и ключа производителя (M_KEY) непосредственно всем зарегистрированным серверам данных, участвующим в схеме защиты.

SA также является хранителем всей документации, касающейся этой части S100. Все рабочие процедуры определяются и регулируются SA.

15-4.2 Серверы данных

Серверы данных (DS) отвечают за шифрование и/или цифровое подписание наборов данных в соответствии с процедурами и процессами, определенными в настоящей части. Серверы данных могут также выдавать лицензии (разрешения на данные - data permits), чтобы Клиенты данных, с действительными разрешениями пользователя (user permits), могли расшифровать данные продукта.

Серверы данных используют информацию M_KEY и M_ID, предоставленную SA, для выдачи зашифрованных ключей продукта для каждой конкретной инсталляции. Даже если ключи, используемые для шифрования каждого набора данных, одинаковы для индивидуальных клиентов данных, они будут зашифрованы с помощью уникального HW_ID и поэтому не могут быть переданы другими системными установками того же производителя.

Схема не препятствует агентам или дистрибьюторам в предоставлении услуг данных своим клиентам. Соглашения и структуры для достижения этой цели выходят за рамки настоящего документа. Настоящий документ содержит только технические спецификации для подготовки защищенных наборов данных, соответствующих этому стандарту.

Гидрографические службы, производители данных, реселлеры с добавленной стоимостью и организации RENC являются примерами серверов данных.

15-4.3 Клиенты данных

Клиенты данных (DC) являются конечными пользователями наборов данных и будут получать защищенную информацию от серверов данных для доступа и использования наборов данных и услуг. Программное обеспечение DC (система OEM) отвечает за аутентификацию цифровых подписей, применяемых к файлам продукта, и расшифровку данных в соответствии с процедурами, определенными в схеме.

Пользователи систем ECDIS/ECS являются примером клиентов данных.

15-4.4 Производители оригинального оборудования

Производители оригинального оборудования (OEMs), подписавшиеся наСхему защиты данных S-100 должны создавать свое программное обеспечение в соответствии со спецификациями, определенными в данном документе, и сами проверять и оценивать его по правилам, установленным SA. Эта часть устанавливает тестовые данные для испытания и проверки приложений OEM, созданных на основе S-100, когда продукты становятся доступными. SA обеспечивает успешно зарегистрированных OEM, уникальным ключом и идентификацией производителя (M_KEY и M_ID).

Производитель должен обеспечить безопасный механизм в своих программных системах для уникальной идентификации каждой инсталляции конечного пользователя. Схема требует, чтобы каждая инсталляция имела уникальный идентификатор оборудования (HW_ID), если Сервер данных не дал согласие на дублирование.

Программное приложение сможет расшифровать ключи продукта в разрешениях данных с помощью HW_ID, хранящихся на жестком или программном носителе, подключенным к или запрограммированном в приложении, чтобы впоследствии расшифровать и распаковать файлы набора данных. Целостность продукта может

быть проверена путем аутентификации цифровой подписи, предоставляемой вместе с файлами набора данных.

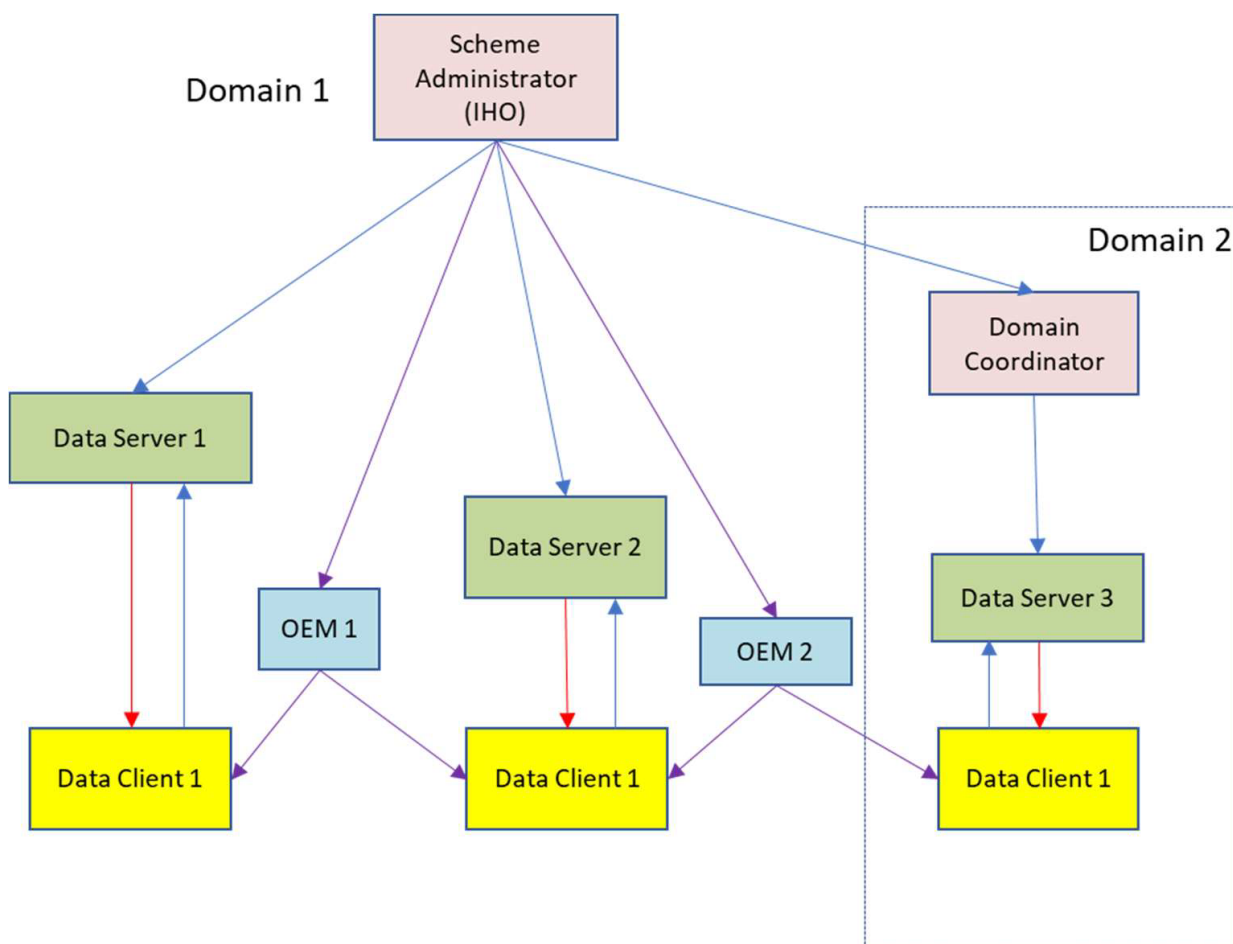
15-4.5 Координатор домена

Координаторы доменов назначаются доверенными органами SA, способными выдавать сертификаты и обеспечивать промежуточную аутентификацию серверов данных в пределах своего домена. Координаторы доменов назначаются SA и имеют делегированные полномочия подписывать сертификаты сервера данных в пределах своего домена. Когда клиенты данных проверяют подлинность цифровых подписей, созданных серверами данных, сертификаты образуют "цепочку" к идентификации корневого уровня SA. Если Сервер Данных сертифицирован Координатором Домена, то Клиент Данных должен также проверить их личность в корневом уровне SA, обеспечивая цепочку аутентификации от набора данных до SA.

15-4.6 Взаимоотношения участников

Администратор Схемы (SA), который может быть только один, проверяет идентичность других участников схемы. Все серверы данных и производители систем (ОЕМ) должны подавать заявку в SA, чтобы стать участниками схемы, и после принятия им предоставляется соответствующая уникальная для них информация. Клиенты данных являются клиентами серверов данных и ОЕМов, где серверы данных предоставляют услуги данных; а ОЕМы оборудование для расширения и отображения этих услуг.

SA подписывает публичный ключ серверов данных для создания их цифрового сертификата, который будет использоваться в работе Схемы защиты. Координаторы доменов также могут подписывать публичные ключи своих организаций-членов для создания своих цифровых сертификатов. Координаторы доменов информируют SA об идентичности сервера данных и их контактных данных. SA распространяет информацию M_ID и M_KEY непосредственно всем серверам данных, участвующим в схеме защиты, когда они присоединятся к схеме, и добавляют клиентов данных.



- ← Поставка M_ID/M_KEY для поддержки защиты данных
- ← Поставка данных конечным пользователям
- ← Аутентификация идентичности

Рисунок 15-1 – Взаимоотношения между участниками схемы защиты

Поскольку схема защиты не зависит от клиентов данных, которые всегда имеют подключение к Интернету для аутентификации сертификатов или для проверки пути сертификата, в метаданные набора обмена данных должно быть включено достаточно информации для выполнения этих функций. Во всех случаях сертификат SA устанавливается на системах конечных пользователей отдельно и не распространяется в метаданных набора обмена для обеспечения независимой проверки сертификата SA.

15-5 Сжатие и пакетирование данных

15-5.1 Обзор

Содержание данных S-100, в силу своей структуры содержит повторяющиеся элементы информации. Примерами этого являются небольшие изменения координат в файле.

Если применяется сжатие, файлы всегда сжимаются до их шифрования, так как эффективность любого алгоритма сжатия зависит от наличия структурированного содержимого данных. Спецификации отдельных продуктов S-100 указывают, используется ли сжатие.

Все файлы набора обмена должны быть подписаны цифровой подписью перед применением сжатия.

15-5.2 Алгоритм сжатия

Схема защиты использует алгоритм ZIP для сжатия и распаковки файлов. Метод сжатия - DEFLATE. Каждый файл сжимается в один файл с тем же именем, что и исходный файл. Если требуется сжать несколько файлов (например, каталог изображений), то они должны быть расположены в одной корневой папке и имя сжатого файла, устанавливается аналогичным имени корневой папки.

Функции шифрования и фичеры цифровой подписи ZIP не используются.

15-5.3 Кодирование

Отдельные спецификации продуктов S-100 будут содержать более подробную информацию, если используется сжатие, и о том какие файлы будут сжиматься.

Использование сжатия кодируется в:

- S-100_ExchangeCatalogue-compressionFlag со значением 1.

15-6 Кодирование данных

15-6.1 Какие данные кодируются?

Любая спецификация продукта на базе модели данных S-100, должна определять, будет ли использоваться шифрование и какие файлы будут зашифрованы.

При шифровании алгоритм шифрования должен быть Advanced Encryption Standard (AES) в режиме Cipher Block Chaining (CBC). Всегда предполагается, что будет зашифрован полный файл.

Кроме того, аппаратный ID системы OEM (HW_ID) будет зашифрован и предоставлен клиенту данных в виде разрешения пользователя (user permit). Ключи, используемые для шифрования файлов, шифруются Сервером Данных и

поставляются Клиентам Данных как разрешения данных (data permits). Информация об алгоритме шифрования дается в пункте 15-6.2.1.

15-6.2 Как они кодируются?

Каждый отдельный продукт шифруется с помощью уникального ключа. Тот же ключ используется для шифрования всех файлов, связанных с продуктом, и всех обновлений, выпущенных для данного издания продукта. Однако схема позволяет изменять ключи по усмотрению сервера данных. Ключи доставляются клиентам данных в виде разрешений на данные (data permits).

15-6.2.1 Алгоритм кодирования

Для шифрования разрешений и файлов данных используется алгоритм блочного шифрования Advanced Encryption Standard (AES). Это алгоритм с симметричным ключом. Это означает, что один и тот же ключ используется для шифрования и расшифровки. Алгоритм определяет, как один блок простого текста преобразуется в один блок шифротекста и наоборот. Размер блока AES всегда составляет 16 байт (128 бит). Длина ключа может быть выбрана из 128 бит, 192бит или 256 бит. Соответствующие варианты называются AES-128, AES-192 или AES-256. В этой части S-100 всегда используется 128-битная длина ключа.

Алгоритм AES может шифровать только один блок обычного текста. Для более крупных сообщений используется режим блочного шифрования. Эта схема защиты выбирает режим цепочного шифрования блоков (CBC) для шифрования более одного блока данных. В этом режиме работы требуется, чтобы длина обычного текста была в точности кратна размеру блока; требуется педдинг.

15-6.2.2 Педдинг кодирования

Используемый метод педдинга описан в PKCS#7. Он добавляет N байт к сообщению до тех пор, пока его длина не увеличится до 16 байт. Значение каждого байта равно N. Обратите внимание, что если длина исходного текста равна 16, то должен быть добавлен полный блок 16 байт, каждый из которых имеет значение 16.

Таблица 15-1 – Педдинг простого текста

Plain text	Padded Plain Text
xx	xx 0F 0F 0F 0F 0F 0F 0F 0F 0F 0F 0F 0F 0F 0F
xx xx	xx xx 0E 0E 0E 0E 0E 0E 0E 0E 0E 0E 0E 0E 0E
xx xx xx	xx xx xx 0D 0D 0D 0D 0D 0D 0D 0D 0D 0D 0D 0D
xx xx xx xx	xx xx xx xx 0C 0C 0C 0C 0C 0C 0C 0C 0C 0C 0C
xx xx xx xx xx	xx xx xx xx xx 0B 0B 0B 0B 0B 0B 0B 0B 0B 0B
xx xx xx xx xx xx	xx xx xx xx xx xx 0A 0A 0A 0A 0A 0A 0A 0A 0A
xx xx xx xx xx xx xx	xx xx xx xx xx xx xx 09 09 09 09 09 09 09 09
xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx 08 08 08 08 08 08 08
xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx 07 07 07 07 07 07
xx xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx xx 06 06 06 06 06
xx xx xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx xx xx 05 05 05 05
xx xx xx xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx xx xx xx 04 04 04
xx xx xx xx xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx xx xx xx 03 03 03
xx xx xx xx xx xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx xx xx xx xx 02 02
xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx 01
xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx	xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx 10 10 10 10 10 10 10 10 10 10 10 10 10 10

xx = Произвольные байты

15-6.2.3 Шифрование AES режим CBC

В режиме CBC каждый блок простого текста - это XORed с предыдущим блоком шифрования простого текста перед шифрованием. Для первого блока требуется вектор инициализации IV. Математическая формула следующая:

$$C_i = E_K(P_i \oplus C_{i-1}); i \geq 1 \quad (3a)$$

$$C_0 = IV \quad (3b)$$

C_i - это блок i^{th} зашифрованного текста; P_i - это блок i^{th} простого текста. E_K - это метод шифрования AES, шифрующий ровно один блок. IV - это вектор инициализации, а \oplus - это операция XOR.

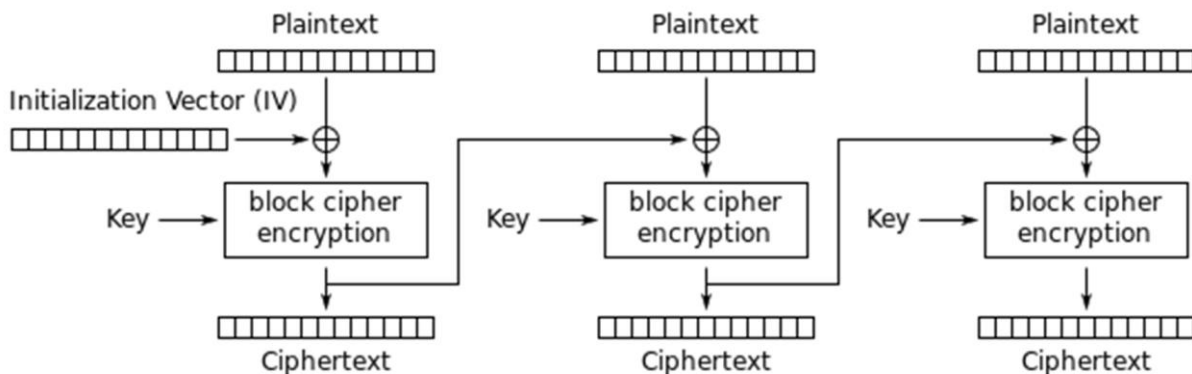


Рисунок 15-2 – Режим шифрования Cipher Block Chaining (CBC) (Источник: Wikipedia)

Дешифрование определяется как:

$$P_i = D_K(C_i) \oplus C_{i-1}; i \geq 1 \quad (4a)$$

$$C_0 = IV \quad (4b)$$

D_K - это метод дешифрования AES, дешифрующий ровно один блок.

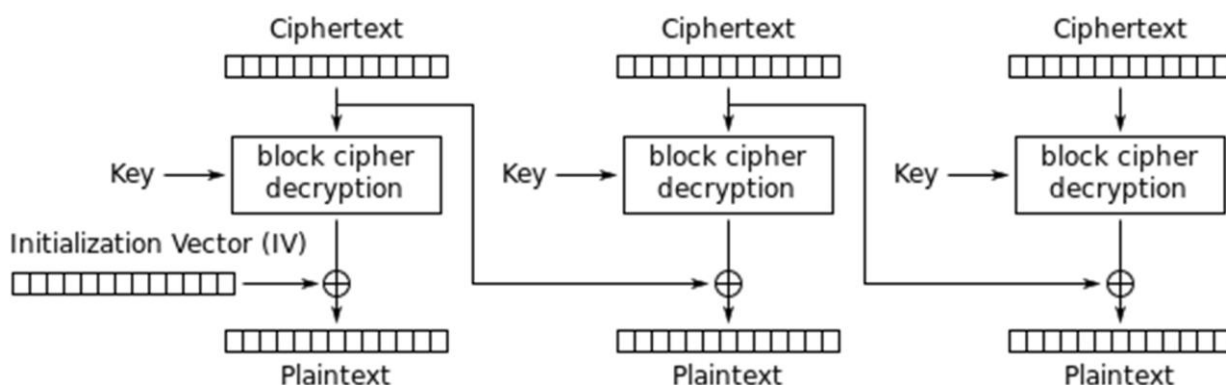


Рисунок 15-3 – Режим дешифрования Cipher Block Chaining (CBC) (Источник: Wikipedia)

15-6.2.4 Режим CBC AES– вектор инициализации

Обычно вектор инициализации должен быть перенесен с шифрования на дешифрование. Однако неправильный IV при расшифровке повредит только первый блок простого текста. Это легко понять из формул и диаграмм. Каждый простой текстовый блок зависит только от двух смежных зашифрованных текстовых блоков.

Это поведение будет использоваться при следующей модификации режима CBC.

При шифровании файлов данных простой текст будет дополнен одним случайным блоком. Затем шифрование выполняется в обычном режиме с использованием случайного вектора инициализации. Этот вектор не должен передаваться на расшифровку клиенту данных.

При расшифровке может использоваться произвольный вектор инициализации, а после нормальной расшифровки CBC первый простой текстовый блок удаляется. Остальное – это файл данных оригинального простого текста.

Эта процедура не требует переноса IV или использования прогнозируемого IV в рамках data permit. Первый вариант усложнит процесс передачи данных, а второй сделает его уязвимым для атак, особенно если первые блоки обычного текста общеизвестны (как ISO/IEC 8211 Data Descriptive Records).

15-6.2.5 Примеры AES

Следующие примеры взяты из документации FIPS.

Шифрование и дешифровка ровно одного блока:

Key₁₂₈: K = {00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 0a, 0b, 0c, 0d, 0e, 0f}
Plain Text: P = {00, 11, 22, 33, 44, 55, 66, 77, 88, 99, aa, bb, cc, dd, ee, ff}
Cipher Text: C = {69, c4, e0, d8, 6a, 7b, 04, 30, d8, cd, b7, 80, 70, b4, c5, 5a}

Key₁₉₂: K = {00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 0a, 0b, 0c, 0d, 0e, 0f,
10, 11, 12, 13, 14, 15, 16, 17}
Plain Text: P = {00, 11, 22, 33, 44, 55, 66, 77, 88, 99, aa, bb, cc, dd, ee, ff}
Cipher Text: C = {dd, a9, 7c, a4, 86, 4c, df, e0, 6e, af, 70, a0, ec, 0d, 71, 91}

Key₂₅₆: K = {00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 0a, 0b, 0c, 0d, 0e, 0f,
10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 1a, 1b, 1c, 1d, 1e, 1f}
Plain Text: P = {00, 11, 22, 33, 44, 55, 66, 77, 88, 99, aa, bb, cc, dd, ee, ff}
Cipher Text: C = {8e, a2, b7, ca, 51, 67, 45, bf, ea, fc, 49, 90, 4b, 49, 60, 89}

Следующий пример документирует модифицированный режим CBC:

Key₁₂₈: K = {12, 34, 56, 78, 9a, bc, de, f0, 12, 34, 56, 78, 9a, bc, de, f0}
Plain Text: P = {fe, dc, ba, 98, 76, 54, 32, 10}

Простой текст после предпеддинга случайного блока:

P' = {48, d2, 4e, 7c, 00, 2f, 67, 4e, 93, 1d, ee, 27, 42, 17, a3, 4c}
{fe, dc, ba, 98, 76, 54, 32, 10}

Простой текст (padded):

P'' = {48, d2, 4e, 7c, 00, 2f, 67, 4e, 93, 1d, ee, 27, 42, 17, a3, 4c}
{fe, dc, ba, 98, 76, 54, 32, 10, 08, 08, 08, 08, 08, 08, 08}

Вектор инициализации (случайный):

IV_E = {45, b5, 00, d7, 28, 39, 42, bb, 85, 61, 28, d5, 97, 15, ca, 25}

Шифрованный текст с использованием режима CBC:

$$C = \{ba, 45, ee, 06, 02, a6, 29, 35, 7a, e3, 90, 2c, 22, 4d, d9, d5\}$$
$$\{dd, 3b, 07, 3b, 84, 7f, 4d, 43, 28, 71, 19, 43, 97, d9, a6, 03\}$$

Для расшифровки может использоваться произвольный вектор инициализации; например:

$$IV_D = \{00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00\}$$

Расшифровка с использованием CBC даст следующий простой текст. Байты, добавленные педдингом, уже удалены:

$$P_D' = \{0d, 67, 4e, ab, 28, 16, 25, f5, 16, 7c, c6, f2, d5, 02, 69, 69\}$$
$$\{fe, dc, ba, 98, 76, 54, 32, 10\}$$

Обратите внимание, что первый блок отличается от блока в P' .

После отбрасывания первого блока исходное сообщение восстанавливается.

$$P_D = \{fe, dc, ba, 98, 76, 54, 32, 10\} = P$$

15-7 Шифрование и лицензирование данных

15-7.1 Введение

Клиенты данных обычно не покупают продукты на базе S-100, а получают лицензию на их использование. Лицензирование - это метод, используемый серверами данных для предоставления клиентам данных избирательного доступа к актуальным продуктам в течение определенного периода времени.

Для эффективной работы схемы должно быть средство, с помощью которого системы клиентов данных могут разблокировать зашифрованные данные. Для разблокировки данных система клиента данных должна иметь доступ к ключам, которые использовались для шифрования лицензированных файлов данных. Эти ключи предоставляются клиенту данных в зашифрованном виде в файле разрешений, содержащем набор разрешений. Именно эти разрешения данных содержат ключи шифрования. Этот метод используется для файлового обмена данными между клиентом данных и сервером данных. В других системах и методологиях, например, в потоковых данных, могут использоваться либо различные алгоритмы, либо различные длины ключей, с указанием в метаданных, как они определяются

Для того, чтобы каждый набор разрешений на данные был эксклюзивным, ключи должны быть зашифрованы с помощью чего-то уникального для системы клиента данных. OEM присваивают идентификатор (HW_ID) каждой из своих систем и предоставляют его зашифрованную копию в виде user permit каждому клиенту данных. HW_ID зашифрован и хранится в user permit.

OEMs шифруют HW_ID своим уникальным ключом производителя (M_KEY), так что HW_ID не может быть повторен другим производителем. ИНО, как Администратор

Схемы, предоставляет серверам данных доступ к M_KEYS OEMов и, следовательно, может расшифровать HW_ID, хранящийся в user permit. Серверы данных шифруют свои ключи наборов данных с производителями HW_ID при производстве набора разрешений на данные.

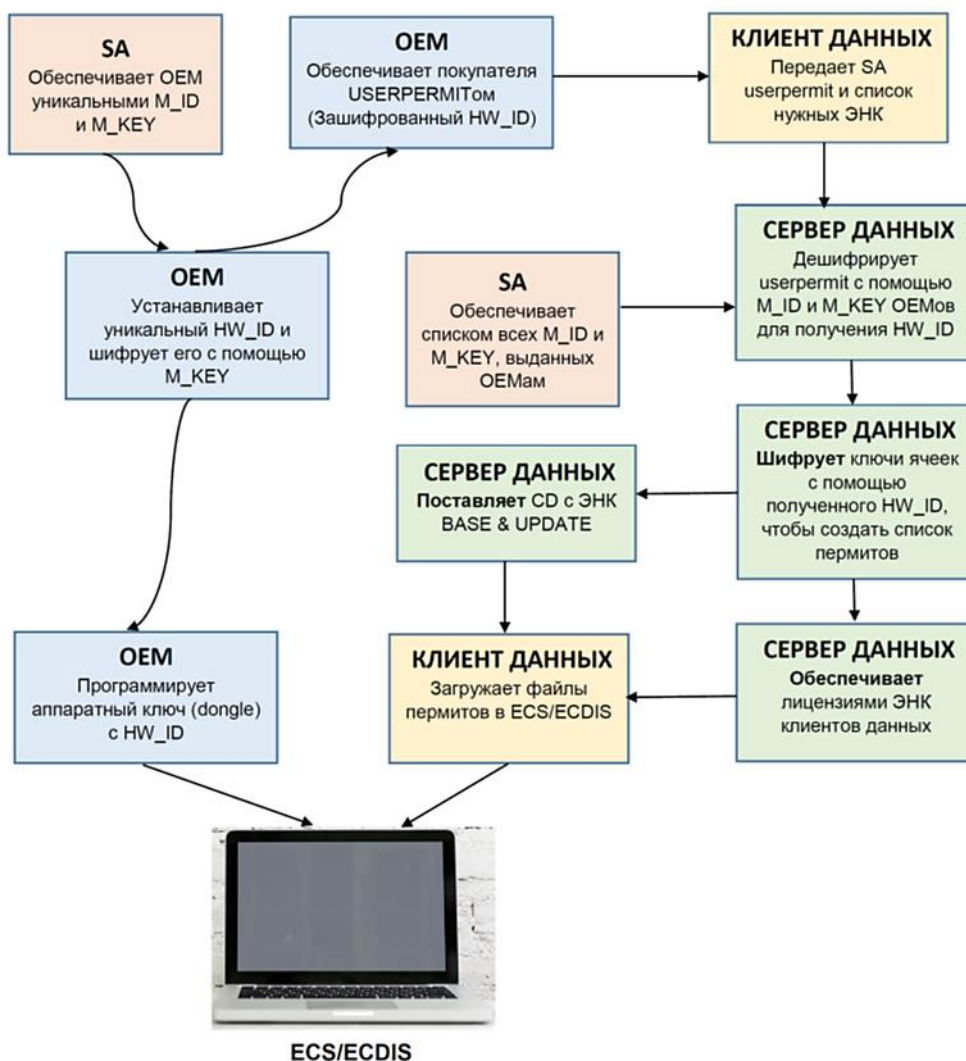


Рисунок 15-4 – Диаграмма лицензирования верхнего уровня на базе ЭНК S-101

15-7.2 Преобразование битовых строк в целые числа

15-7.2.1 Преобразование битовой строки в целое число

Последовательность бит $\{b_1, b_2, \dots, b_n\}$ определяет без знаковое целое число, как:

$$I = b_1 2^{n-1} + b_2 2^{n-2} + \dots + b_{n-1} 2^1 + b_n; b_i \in \{0,1\} \quad (1a)$$

или

$$I = \sum_{i=1}^n b_i 2^{n-i} \quad (1b)$$

Бит b_1 является самым значимым битом, а бит b_n является наименее значимым битом в последовательности. Целое число будет в диапазоне: $0 \leq I < 2^n$.

В большинстве реализаций битовая строка будет организована как последовательность байт $\{B_0, B_1, \dots, B_m\}$, с:

$$B_{m-j} = \{x_{n-8j-7}, x_{n-8j-6}, \dots, x_{n-8j}\}; \forall j \in \{0 \dots m\} \text{ with } x_i = \begin{cases} b_i; & \forall i \ i > 0 \\ 0; & \forall i \ i \leq 0 \end{cases} \text{ and } m = \left\lceil \frac{n}{8} \right\rceil \quad (2)$$

Возможная реализация преобразования такой байтовой последовательности в целое число задаётся следующим псевдокодом.

Вход: Последовательность байт $B = \{B_0, B_1, \dots, B_m\}$

Выход: не отрицательное целое число I

```

Let I=0
for k from 0 to m
    I = I * 28
    I = I + Bk
Return I

```

15-7.2.2 Преобразование целого числа в битовую строку

Формула 1a и 1b описывают, как битовая строка связана с соответствующим (неотрицательным) целым числом. Предполагая, что битовая строка организована как последовательность байт, как определено в (2), следующий алгоритм показывает, как преобразовать беззнаковое целое число в битовую строку.

Input: a non-negative integer number I with $0 \leq I < 2^n$

Output: a sequence of bytes B of length $m = \begin{cases} 1; & I = 0 \\ \lceil \frac{n}{8} \rceil; & I > 0 \end{cases}$

Let B be an empty sequence

If I = 0

Append the byte b=0 to B

Else

While I > 0 do

Let c = I mod 2⁸

Prepend c to B

Let I = I div 2⁸

While the length of B is < m

Prepend 0 to B

Return B

Обратите внимание, что деление на 2^8 эквивалентно операции битового сдвига $I \gg 8$.

15-7.2.3 Преобразование беззнакового целого числа в шестнадцатеричное текстовое представление.

Следующий псевдокод показывает, как конвертировать беззнаковое целое число в его шестнадцатеричное текстовое представление. В этом текстовом представлении каждая цифра может иметь 16 различных значений.

Целое число I определяется как:

$$I = d_n 16^{n-1} + d_{n-1} 16^{n-2} + \dots + d_2 16 + d_1$$

Таблица 15-2 – Преобразование беззнакового целого числа в шестнадцатеричный текст

Digit d	Bit string	Character	ASCII Code (Hex)	ASCII Code (dec)
0	0000	'0'	30	48
1	0001	'1'	31	49
2	0010	'2'	32	50
3	0011	'3'	33	51
4	0100	'4'	34	52
5	0101	'5'	35	53
6	0110	'6'	36	54
7	0111	'7'	37	55
8	1000	'8'	38	56
9	1001	'9'	39	57
10	1010	'A'	41	65
11	1011	'B'	42	66
12	1100	'C'	43	67
13	1101	'D'	44	68
14	1110	'E'	45	69
15	1111	'F'	46	70

Алгоритм следующий:

Input: An unsigned integer number I

Output: The hexadecimal text representation S

Let S be an empty sequence of characters.

If I = 0

Let S = "0"

Else

While I > 0

Let c be the character corresponding to the value $d = I \bmod 16$

Prepend c to S

Let $I = I \div 16$

Return S

15-7.2.4 Преобразование шестнадцатеричного текста в целое число без знака

Следующий алгоритм показывает, как преобразовать шестнадцатеричное текстовое представление беззнакового целого числа в само целое число.

Input: A hexadecimal text representation S of an unsigned integer number $S = \{s_1, s_2, \dots, s_m\}$

Output: An unsigned integer number I

Let $I = 0$

For $I = 1$ to m

*$I = I * 16$*

$I = I + d$; where d is the digit value corresponding to the character S_i

Return I

15-7.3 User Permit (Разрешение пользователя)

User permit создается OEMом и поставляется клиентам данных вместе с системой для того, чтобы они могли получить необходимый доступ к зашифрованным продуктам от серверов данных. В следующем разделе определяются состав и формат user permit.

Все клиенты данных с системами, способными использовать данные, защищенные в соответствии со схемой защиты данных МГО, должны иметь идентификатор оборудования (HW_ID), встроенный в систему конечного пользователя. Такой идентификатор HW_ID часто реализуется в виде ключа или другим способом, обеспечивающим стабильную идентификацию каждой инсталляции.

HW_ID не известен Клиенту данных, но OEM предоставляет ему пользовательское разрешение, которое является зашифрованной версией HW_ID и уникально для каждой системы Клиента данных. Отметим, что user permit создается путем шифрования HW_ID ключом производителя (M_KEY). Алгоритм CRC32 применяется к зашифрованному HW_ID и результат прилагается к нему. В заключение, производитель прикрепляет свой идентификатор производителя (M_ID) в конце полученной строки. Значения M_KEY и M_ID поставляются Администратором схемы и уникальны для каждого производителя, предоставляющего системы, совместимые со схемой защиты данных ИНО.

Клиент данных получает доступ к зашифрованным продуктам на основе S-100, предоставляя свой user permit серверу данных. Это позволяет серверу данных выдавать разрешение на данные (Data Permit), соответствующие представленному user permit. Поскольку в user permit содержится уникальный M_ID производителя, сервер данных использует это для определения, какой M_KEY использовать для расшифровки ID оборудования, содержащегося в user permit. M_ID указывается шестью последними символами в user permit. Список значений M_KEY и M_ID производителей выпускается и обновляется Администратором схемы (SA) для всех серверов данных, подписавшимся на схему. Этот список периодически обновляется по мере присоединения к схеме новых OEMов.

15-7.3.1 Определение user permit

user permit состоит из 28 символов и записывается как текст ASCII со следующим обязательным форматом и длиной полей:

Таблица 15-3 – Структура поля user permit

Зашифрованный HW_ID	Контрольная сумма SUM (CRC)	ID производителя M_ID
128 бит (32 шест. цифр)	8 шестнадцатеричных цифр	6 шестнадцатеричных цифр

Любой алфавитный символ записывается в верхнем регистре (прописными буквами).

Пример: структура user permit:

AD1DAD797C966EC9F6A55B66ED98281599B3C7B1859868

Структура этого user permit объясняется в следующих подразделах.

15-7.3.1.1 Формат HW_ID

HW_ID - это шестнадцатеричное число из 32 цифр, определяемое OEMом. Такой HW_ID может быть реализован в виде ключа или другим способом, обеспечивающим идентификацию и защиту от несанкционированного доступа каждой инсталляции.

HW_ID хранится в зашифрованной форме в user permit. Он зашифрован с помощью алгоритма AES с M_KEY в качестве ключа, представляющее 128 битовое значение, закодированное как 32 символьное (16 байт) шестнадцатеричное число и использующее IV из 16 нулевых байт (используются фиксированные IV, чтобы избежать необходимости включения их в user permit). Закодированный HW_ID далее представляется в форме ASCII в user permit как 32 цифры.

Отметим, что размер HW_ID идентичен размеру блока AES и не требует никакого педдинга.

Пример HW_ID: **40384B45B54596201114FE9904220101**

Пример зашифрованного HW_ID: **AD1DAD797C966EC9F6A55B66ED982815**

(M_KEY= **4D5A79677065774A7343705272664F72**)

15-7.3.1.2 Формат контрольной суммы (CRC)

Контрольная сумма представляется шестнадцатеричным числом из 8 цифр. Она генерируется путём преобразования зашифрованного HW_ID в шестнадцатеричную строку с 32 символами. Затем она хешируется с использованием алгоритма CRC32 и 4 байта преобразовываются в шестнадцатеричную строку из 8 символов.

Контрольная сумма не шифруется и позволяет проверить целостность user permit.

В примере выше контрольная сумма это:

- Пример HW_ID: **40384B45B54596201114FE9904220101**
- Пример зашифрованного HW_ID: **AD1DAD797C966EC9F6A55B66ED982815**
- Контрольная сумма: **99B3C7B1**

15-7.3.1.3 Формат M_ID

M_ID - это 6-символьный буквенно-цифровой код, выраженный в ASCII, предоставляемый Администратором схемы (SA). SA предоставляет всем лицензированным производителям систем их собственную уникальную комбинацию ключа и идентификатора производителя (M_KEY и M_ID). Производитель системы должен обеспечить защиту этой информации.

SA предоставляет всем лицензированным серверам данных полный список всех кодов производителя, как только новые производители подписываются на схему. Эта информация используется сервером данных для определения ключа (M_KEY), который будет применяться для расшифровки HW_ID в user permit при создании data permit набора данных, заказанных клиентом данных.

В выше приведенном примере M_ID это: **859868**

15-7.3.2 Формат M_KEY

M_KEY — это случайное шестнадцатеричное (128 битное) число из 16 байт, присвоенное производителю системы и предоставляемой Администратором схемы. OEM использует этот ключ для шифрования значений HW_ID при генерации user permit. Этот ключ также используется сервером данных для расшифровки HW_IDs. Отметим, что размер M_KEY идентичен размеру блока AES и не требует никакого педдинга.

Пример M_KEY: **4D5A79677065774A7343705272664F72** (Шестнадцатеричное представление)

Полный пример показан в таблице 15-4 ниже:

Таблица 15-4 – Полный user permit – пример

Field	Value
M_ID	859868
M_KEY	4D5A79677065774A7343705272664F72
HW_ID	40384B45B54596201114FE9904220101
Encrypted HW_ID	AD1DAD797C966EC9F6A55B66ED982815
CRC32 (Encrypted HW_ID)	99B3C7B1
Complete User Permit	AD1DAD797C966EC9F6A55B66ED98281599B3C7B1859868

15-7.4 Data Permit (Разрешение на данные)

Для расшифровки файла данных Клиент Данных должен иметь доступ к ключу шифрования (см. раздел 15-6.2.1), используемого при их шифровании. Поскольку ключи шифрования известны только серверу данных, необходимо предусмотреть средства для передачи этой информации клиентам данных защищенным образом. Эта информация предоставляется сервером данных клиенту данных в зашифрованной форме, известная как permit (разрешение). Для доставки разрешения на данные используется файл с именем PERMIT.XML (см. раздел 15-7.4.1). Этот файл может содержать несколько разрешений, для представления нескольких наборов данных, требуемых клиентом данных.

Файл PERMIT.XML поставляется либо на жестких носителях, либо через онлайн-сервисы в соответствии с операционными процедурами серверов данных. Эти процедуры будут доступны клиенту данных после покупки лицензии.

Каждая запись в файле data permit также содержит дополнительные поля, которые предоставляются, чтобы помочь системам OEMов управлять лицензиями клиентов данных и разрешить использование файлов из нескольких серверов данных, см. раздел 15-7.4.2.

Клиенты данных могут получить лицензию на доступ к продуктам, предоставляя серверу данных свой user permit (пункт 15-7.3). Затем серверы данных извлекают HW_ID из user permit, используя ключ M_KEY клиента данных, и создают permit для клиента на основе этого ключа. Формат файлов permit описывается ниже в пунктах 15-7.4.1-15-7.4.4.

Поскольку data permits выдаются для конкретного HW_ID, они не могут передаваться между инсталляциями (Системами клиентов данных). Этот метод привязки permits к инсталляциям поддерживает производство общих зашифрованных данных, которые могут распространяться среди всех клиентов данных, подписавшихся на услугу.

Система клиента данных расшифровывает permit, используя свой HW_ID, хранящийся с помощью аппаратных или программных средств. Расшифрованные ключи могут затем использоваться системой для расшифровки лицензированных продуктов. Поскольку несколько серверов данных могут создавать файлы permit для определенного типа продуктов, ответственность за управление файлами permit от нескольких серверов данных несет клиент данных.

15-7.4.1 Файл Permit (PERMIT.XML) (Разрешение)

Имя файла всегда представляется ПРОПИСНЫМИ буквами, как и любые алфавитные символы, содержащиеся в файле. Файл полностью кодируется в ASCII. OEMы должны быть осведомлены о том, что все текстовые файлы ASCII, генерируемые Схемой защиты, могут содержать неоднозначные маркеры конца строки, такие как CR или CRLF, и должны быть в состоянии работать с ними.

Структура схемы XML проиллюстрирована на рисунке 15-5 ниже.

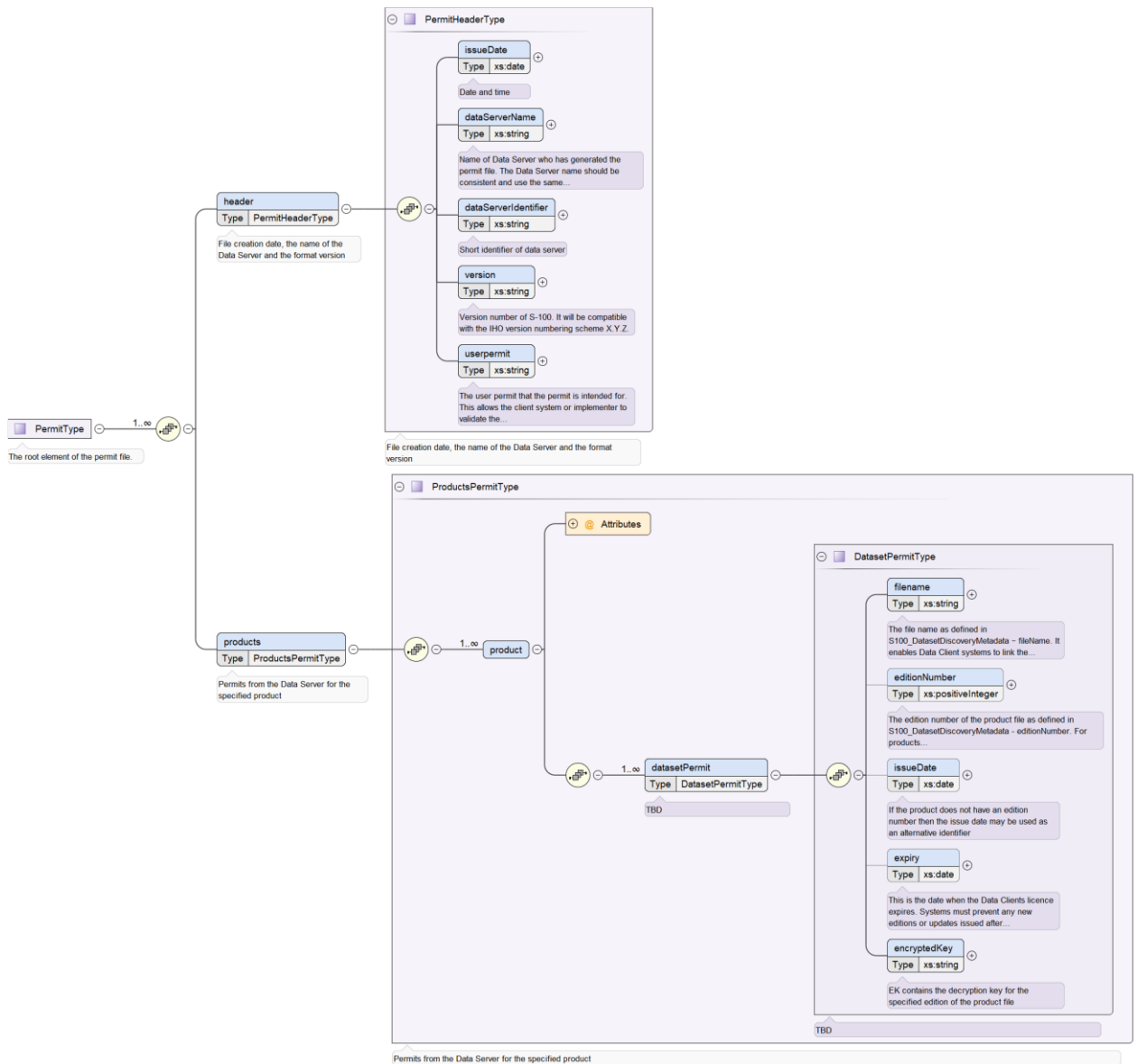


Рисунок 15-5 – Структура файла permit

Файл PERMIT.XML может содержать множество секций с соответствующим элементом XML как указано ниже:

Таблица 15-5 – Элементы PERMIT.XML

Элемент XML	Описание
Header (заголовок)	Дата создания файла, имя сервера данных и версия формата
Products (продукты)	Permits сервера данных для указанного продукта

Обратите внимание, что файл PERMIT.XML может содержать разрешения для нескольких продуктов, предоставленных сервером данных. OEMы должны убедиться, что их программное обеспечение конечного пользователя может объединять разрешения от нескольких серверов данных.

15-7.4.2 Файл Permit – Содержание заголовка

В следующей таблице определяется содержание и формат каждого раздела XML-файла разрешения.

Таблица 15-6 – Содержание и формат PERMIT.XML

Содержание	Элемент XML	Описание
Имя файла	filename	Имя ресурса, для которого предназначено разрешение, без указания пути Формат: Character string
Дата и время	issueDate	Дата Формат XML: xs:date Пример: <issueDate>2018-03-20Z</issueDate>
Провайдер	dataserverName	Имя сервера данных, создавшего файл разрешений. Имя сервера данных должно быть последовательным и использовать тот же организационный контакт, как определено в S100_ExchangeCatalogue - contact Формат XML: xs:string
Версия	version	Номер версии S-100. Он должен быть совместим с системой нумерации версий ИНО X.Y.Z., например, 4.0.0. Формат: Character string
Разрешение пользователя	userpermit	user permit, для которого предназначено разрешение. Это позволяет клиентской системе или установщику проверить назначение. Система конечного пользователя должна быть способна проверить наличие разрешения для указанной системы на многосистемном мосту. Строка символов как определено в пункте 15-7.3.1 Формат: Character string

15-7.4.3 Секции продукта и поля записей разрешения

За элементом заголовка в файле PERMIT.XML следует один элемент под названием " products ", который содержит несколько записей " products ", каждая из которых содержит фактические разрешения на эти продукты. Это позволяет одному файлу PERMIT.XML содержать разрешения для нескольких продуктов, предназначенных для системы конечного пользователя. Атрибут "id" для каждой секции продукта содержит идентификатор S-100 Спецификации продукта, к которой относится продукт; например, <product id="S-101">. Файлы разрешений могут содержать несколько пар элементов заголовков/продукт, относящихся к разным системам конечных пользователей.

15-7.4.4 Определение записи разрешения (Permit Record)

Каждый элемент продукта в файле PERMIT.XML содержит последовательность элементов "permit". Эти элементы содержат фактические разрешения на идентифицированные продукты. В приводимой ниже таблице определяются элементы, содержащиеся в разрешениях, с определением цели каждого из них. Обратите внимание, что разрешения выдаются только для Базовых наборов данных и такое же разрешение используется для расшифровки дополнительных обновлений (если в спецификации продукта реализованы обновления).

Таблица 15-7 – Элементы записи разрешения

Поле	Назначение	Формат
filename	Имя файла, как определено в S100_DatasetDiscoveryMetadata – filename. Это позволяет системам клиентов данных связать правильный ключ шифрования с соответствующим зашифрованным файлом. Путь pathName к файлу определяется в метаданных набора данных.	Character string (строка символов)
editionNumber	[Опция] Номер издания файла продукта, определенный в S100_DatasetDiscoveryMetadata - editionNumber Для продуктов без номера издания разрешение будет применяться ко всем выпущенным наборам данных	Character string
issueDate	[Опция] Если продукт не имеет номера издания, дата выпуска может использоваться в качестве альтернативного идентификатора	xs:date
expiry	Дата истечения срока действия лицензии клиента данных. Системы должны предотвращать установку любых новых изданий или обновлений, выпущенных после этой даты	xs:date
encryptedKey (ЕК)	ЕК содержит ключ для расшифровки указанного издания файла продукта;	32-символьная шестнадцатеричная строка, представляющая 128-битный ключ шифрования

15-7.4.5 Подписи файла Permit

Каждый файл разрешения должен иметь цифровую подпись, созданную сервером данных. Цифровая подпись хранится в отдельном файле и повторно используется в качестве имени файла разрешений, но с добавлением ".SIGN", например, permit.sign.

Содержимое файла подписи представляет собой сертификат сервера данных и подпись файла разрешения, и кодируется в соответствии со схемами S-100 XML. Система OEM должна аутентифицировать сертификат сервера данных до проверки подлинности файла разрешения до расшифровки ключей доступа к набору данных.

15-7.4.6 Пример файла PERMIT.XML

```
<?xml version="1.0" encoding="UTF-8"?>
<Permit xmlns="http://www.iho.int/s100/se/5.1"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.iho.int/s100/se/5.1
https://schemas.s100dev.net/schemas/S100/5.1.0/S100SE/20230327/Part15.xsd">
  <header>
    <issueDate>2018-03-20Z</issueDate>
    <dataServerName>Primar</dataServerName>
    <dataServerIdentifier>PR</dataServerIdentifier>
    <version>1.0.0</version>
  </header>
  <userpermit>267C3AD506E69B1ED18AA5ECC7FFDE6E7C330CE8859868</userpermit>
  <products>
    <product id="S-101">
      <datasetPermit>
        <filename>101GB40079ABCDEF.000</filename>
        <editionNumber>10</editionNumber>
        <expiry>2022-12-31</expiry>
        <encryptedKey>2E16E07E451FF1854156634DA3DD3FB8</encryptedKey>
      </datasetPermit>
      <datasetPermit>
        <filename>101NO32802411223.000</filename>
        <editionNumber>5</editionNumber>
        <expiry>2022-06-10</expiry>
        <encryptedKey>C714B5C0FBDF14BFE4B1F12E62CE5FF6</encryptedKey>
      </datasetPermit>
    </product>
    <product id="S-102">
      <datasetPermit>
        <filename>102NO329048208.h5</filename>
        <editionNumber>1</editionNumber>
        <expiry>2022-12-31</expiry>
        <encryptedKey>50BBC28B6793E1C3966B45FB2932E1BE</encryptedKey>
      </datasetPermit>
    </product>
  </products>
</Permit>
```

15-8 Аутентификация данных

В этом разделе указываются механизмы, структуры и содержание, необходимые для обеспечения защиты данных от копирования и/или методов аутентификации спецификаций продукта S-100. Он определяет стандартизированные методы шифрования файловых компонентов наборов данных, а также каталогов фичеров и изображений. Определяются алгоритмы и методы применения цифровой

подписи, а также соответствующей инфраструктуры, необходимой для управления ключами и обеспечения идентификации в рамках системы защиты данных ИНО.

15-8.1 Введение в проверку подлинности и целостности данных

В технологии цифровой подписи S-100 использует стандартный алгоритм и широко доступный и используемый механизм обмена ключами. Цифровые подписи используют асимметричные алгоритмы публичного ключа в инфраструктурной схеме, подобной PKI, для неразрывной привязки файла данных к личности пользователя.

Схема основывается на ассиметричном шифровании¹ контрольной суммы файла данных. Путем проверки подписи по публичному ключу пользователя, а также путем проверки публичного ключа эмитента на личность высшего уровня, пользователь удостоверяется в личности подписавшегося. Подробное техническое описание цифровых подписей выходит за рамки настоящего документа, и читатель может обратиться к Стандарту цифровой подписи (DSS – FIPS Publication 186) для получения более подробного и доступного объяснения. Эта часть S-100 предполагает базовые знания терминов цифровой подписи и функционирования схем аутентификации PKCS.

Схема защиты данных ИНО может рассматриваться как состоящая из трех фаз:

- 1) Администратор схемы (SA) верифицирует идентичность Сервера данных продуктов S-100 и обеспечивает поставщиков информацией, позволяющей им подписывать их продукты цифровой подписью.
- 2) Сервер данных выпускает продукты, подписанные своим идентификатором, сертифицированным SA.
- 3) Последующая верификация Клиентом данных идентичности Сервера данных, его связи с SA и интеграции в продукт.

Координатор домена может также выступать в роли посредника между Сервером данным и SA. SA сертифицирует подлинность Координатора домена, который, в свою очередь, может сертифицировать подлинность Серверов данных, за которые он несет ответственность.

Следует отметить, что механизм цифровой подписи S-100 предназначен не только для файлов данных спецификаций продукции S-100. Можно шифровать (и выдавать разрешения) и в цифровом виде подписывать любые файловые данные и механизмы, описанные в этой части, каталоги и другие дополнительные файлы, включая Каталоги фичеров и изображений.

¹ Асимметричная криптография опирается на алгоритмы, в которых шифрование и расшифровка происходят различными криптографическими ключами. Поэтому один человек может зашифровать данные и сделать доступным ключ расшифровки для других, чтобы расшифровать его. Эти ключи называются "частным ключом" (private key) и "публичным ключом" (public key), которые в совокупности называются "парой ключей".

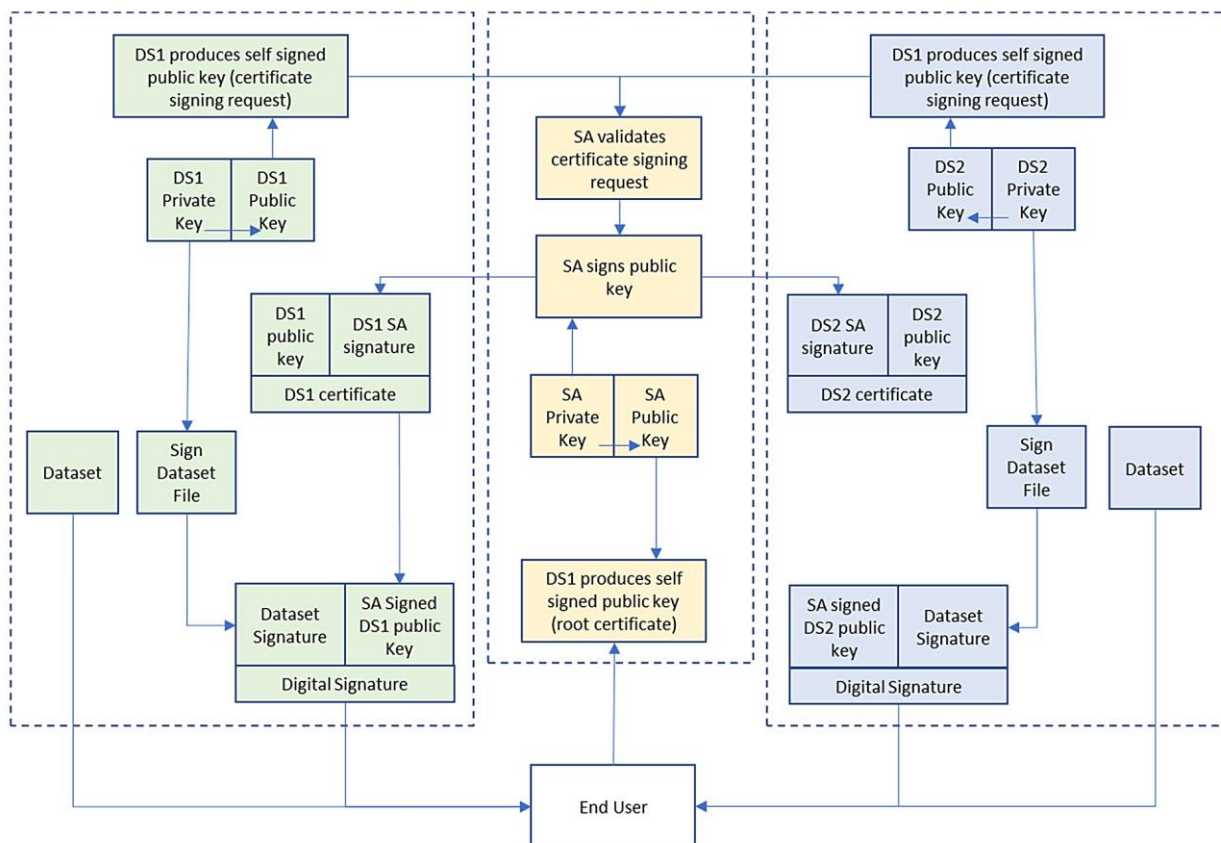


Рисунок 15-6 – Процесс создания сервера данных и цифровой подписи

15-8.2 Настройка схемы защиты данных, регистрация сервера данных и последовательность аутентификации

Ниже приводится перечень шагов, предпринимаемых каждым органом в рамках Схемы защиты данных в ходе цифрового подписания файлов данных.

1. Создание и настройка схемы (только единожды, по инициативе схемы защиты данных):
 - a. SA создает свою собственную пару публичный/частный ключей и самоподписывает ее.
 - b. SA выкладывает свой самоподписанный публичный ключ (также известный как “сертификат”) в публичный домен.
 - c. Публичный ключ SA встраивается в системы OEMов.
2. Установка сервера данных (только один раз):
 - a. Сервер данных создает пару публичный/частный ключей.
 - b. Сервер данных подписывает свой публичный ключ (со своим частным ключем), создавая самоподписанный ключ (также иногда называемый “certificate signing request” (запрос на подпись сертификата)).

с. Самоподписанный ключ сервера данных (SSK) отсылается SA для проверки при подаче заявки на присоединение к Схеме защиты данных ИНО S-100. Любые другие требования и обязанности в рамках схемы защиты данных на данном этапе передаются потенциальному серверу данных.

3. Проверка подлинности сервера данных:

а. В случае принятия SA проверяет SSK и подлинность сервера данных.

б. SA подписывает SSK сервера данных своим частным ключом, чтобы создать сертификат сервера данных, подписанный SA.

в. Далее сертификат сервера данных возвращается серверу данных.

г. Сервер данных проверяет публичным ключем SA, что сертификат подписывает его публичным ключем.

4. Сервер данных может затем создавать цифровые подписи файлов данных. По мере необходимости участники схемы могут также производить цифровые подписи каталогов объектов и изображений.

15-8.3 Проверка цифровых подписей

Проверка цифровых подписей системой клиента состоит из следующих этапов:

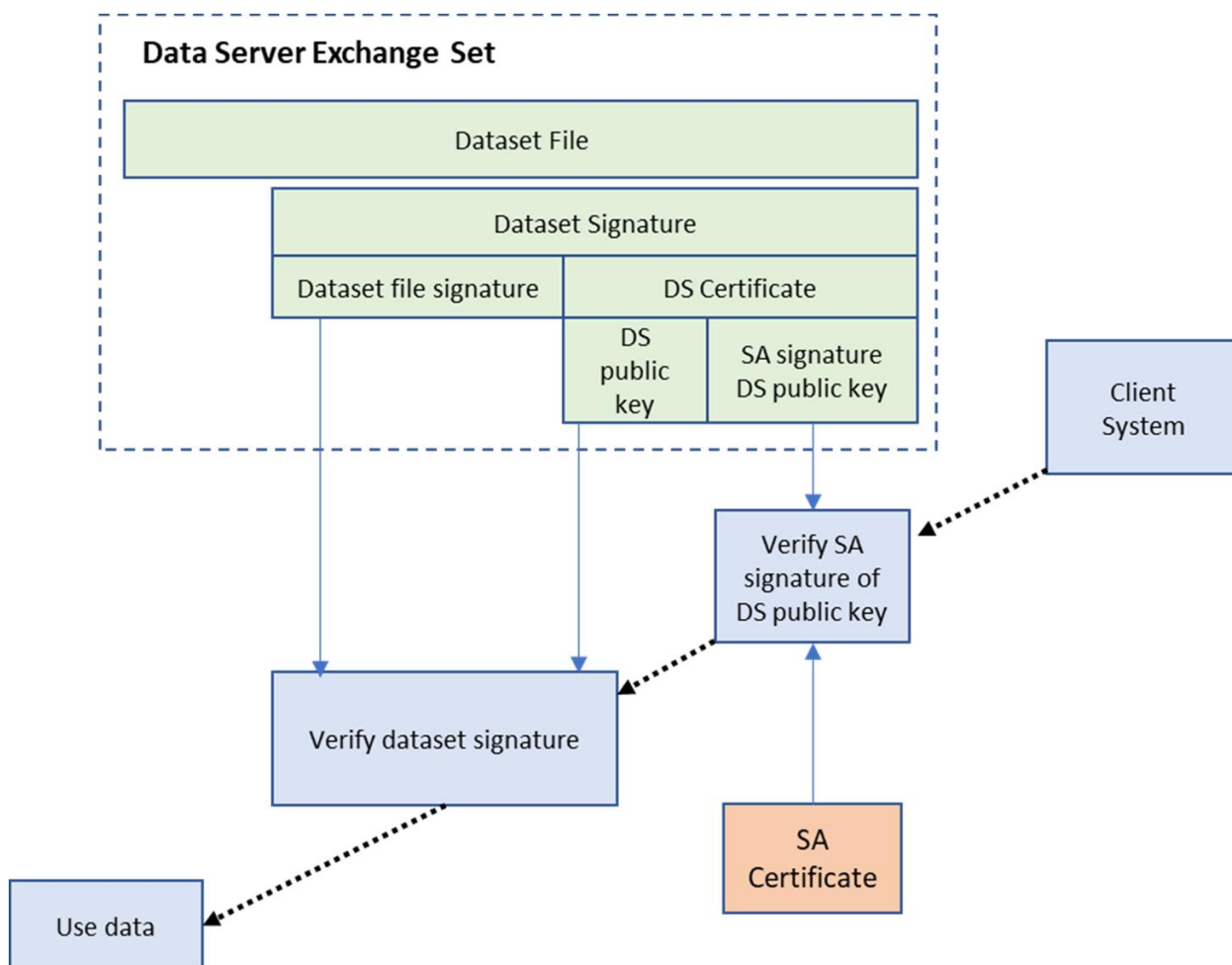


Рисунок 15-7 – Процесс проверки подлинности данных системой клиента.

15-8.4 Форматы данных и стандарты цифровых подписей, ключей и сертификатов.

Следующие категории данных требуют аутентификации:

1. Пары ключей, Частный и Публичный ключи. Это все, закодированные по алгоритму PEM (Elliptical Curve Digital Signature Algorithm (ECDSA)) ключи, вместе с их параметрами домена эллиптической кривой. Форматы описаны в RFC 5915 (Частные ключи) и RFC 5480 (Формат идентификатора и публичного ключа). «Неограниченный» идентификатор алгоритма должен быть установлен на:

```
id-ecPublicKey OBJECT IDENTIFIER ::= {
  iso(1) member-body(2) us(840) ansi-X9-62(10045) keyType(2) 1 }
```

Параметр `namedCurve` должен быть установлен на:

```
secp384r1 OBJECT IDENTIFIER ::= {
  iso(1) identified-organization(3) certicom(132) curve(0) 34 }
```

2. Запросы на подписи сертификатов и цифровые публичные ключи. Когда публичный ключ самоподписан в цифровом виде, он называется "сертификатом" (поскольку публичный ключ "сертифицирован" с помощью частного ключа для его аутентификации). Когда публичный ключ подписан

соответствующим частным ключом, он называется "самоподписанным" сертификатом. Они представлены в виде X.509 записей и могут быть закодированы в DER или PEM для отправки в SA для подписания. При встраивании в XML-файлы ключи должны быть закодированы в PEM так, чтобы простой текст мог быть вставлен как XML-элемент. Для соответствия этому стандарту CSR и сертификаты должны определять алгоритм хеширования, который будет использоваться для создания цифровых подписей к SHA384 (также известным как SHA2-384).

3. Цифровой формат подписанных SA публичных ключей ("сертификатов") является форматом X509v3, закодированный как PEM.

Отличительное имя (DN) в сертификате X.509 является частью неизменного содержимого сертификата (то есть оно не может быть изменено без признания сертификата недействительным). Роли участников схемы и домены, которым они назначены, могут быть закодированы в DN. Оперативные процедуры ИЮ для схемы защиты данных предусматривают любые конкретные процедуры, необходимые для форматирования этого содержания. SA может устанавливать ограничения на значения, разрешенные в компонентах DN (например, общее название или организация), и формат таких идентификаторов для управления функционированием схемы защиты данных среди ее участников.

Политика и процедуры, осуществляемые SA, не входят в сферу компетентности настоящей части S-100 и определяются в других разделах. Использование DN для определения полей certificateRef может также помочь реализаторам в выборе правильного сертификата при проверке цифровой подписи. Это может быть также санкционировано SA, поскольку в нем указывается, как работает Схема защиты данных.

Формат PEM определяет текстовое кодирование множества больших чисел, требуемых алгоритмом ECDSA (вместе с параметрами ECDSA, требуемыми алгоритмом ECDSA). Кодирование PEM (первоначально разработанное для кодирования электронной почты, но широко используемое в сообществе шифрования для кодирования длинных целых чисел, используемых для ключей и цифровых подписей) позволяет встраивать публичные ключи и сертификаты сервера данных в XML-файлы для создания XML-файлов разрешений, создания метаданных каталога и файла поддержки и производства цифровых подписей каталогов изображений и фичеров. Цифровые подписи файлов данных S-100 должны быть встроены в метаданные каталога и выполнять двойную функцию контрольной суммы в отношении незашифрованного файла данных и аутентификации его источника. Поэтому они должны быть изготовлены до использования любого механизма шифрования, так как защита от копирования сама по себе необязательна.

Сертификат SA представляет собой публичный ключ ECDSA, предоставленный, как указано, в виде PEM-файла. Сертификат SA всегда доступен в виде файла ИЮ.PEM. Файл ИЮ.PEM можно получить на сайте ИЮ <http://www.ihp.int>.

Цифровые подписи S-100 реализуются по стандарту цифровой подписи (Digital Signature Standard (DSS)). DSS использует утвержденную хеш-функцию для создания резюме (хэша) содержимого файла. Дайджест сообщения затем вводится

в Elliptic Curve Digital Signature Algorithm (ECDSA), чтобы генерировать цифровую подпись для сообщения с помощью асимметричного алгоритма шифрования и 'Private Key' пары ключей подписчика. Аутентификация на основе файла S-100 использует кривую NIST P-384 и функцию хеширования SHA384. Другие системы или потоки данных через API могут использовать различные кривые и функции хеширования.

В алгоритме ECDSA подпись — это последовательность двух целых чисел. По соглашению они называются R и S (пара "R, S"). Формат цифровых подписей при встраивании в файлы XML выглядит следующим образом:

```
<digitalSignature id="primar" certificateRef="root">
MGQCMDP17NEJXU7gzwTQAp2lgyDzJdlagCeoz6FZOMGFRmV4sPfzAUh1C3hdj+DF
3n2n/QIwPYzh15YiBgJ5Aph11kFUjLywzjDZGHYm/GyjxeCL/8FnOviMwccTlxh6
5fNkL0eg==</digitalSignature>
```

Кодировка двух целых чисел R,S является байтовой последовательностью Base64 ASN.1². Они производятся изначально реализацией openssl и могут генерироваться и проверяться без необходимости распаковывания отдельных чисел R и S. Эта кодировка удобно переносит два значения однозначно в байтовый массив. Последовательность ASN.1, представляющая пару R,S, затем кодируется Base64 (RFC 4648) для представления в элементах цифровой подписи XML.

Схема ASN.1 для примера выше следующая:

SEQUENCE (2 elem)

INTEGER (382 bit)

[33F5ECD1095D4EE0CF04D0029DA58320F325DD5A8027A867A15938C185466578B0F7F30148650B785D8FE0C5DE7DA7FD](#)

INTEGER (382 bit)

[3D8CE1D79622060279029875D641548CVCB0CE30D9187626FC6A3C5E08BFFC1673AF88CC1C71397187AE5F3642F47A0](#)

Цифровая подпись также содержит следующие атрибуты:

1. Атрибут "id" служащий идентификатором.
2. Атрибут certificateRef идентифицирующий сертификат dataserver с правильным публичным ключем в нем для аутентификации. Если подпись удостоверена SA, то certificateRef является идентификатором SA, определенным в элементе schemeAdministrator типа контейнера XML.

Эти атрибуты описаны в разделе 15-8.8.

² Abstract Syntax Notation One (ASN.1) – это описательный язык стандартного интерфейса для определения структур данных, которые могут быть сериализованные и десериализованные межплатформенным образом. Он широко используется в телекоммуникациях и компьютерных сетях, особенно в криптографии. https://en.wikipedia.org/wiki/Abstract_Syntax_Notation_One

15-8.5 Создание ключевых материалов и запросов на подписание сертификатов (подписанные публичные ключи)

Используемый всеми "пакет openssl" обеспечивает публичный домен, инструмент с открытым исходным кодом для производства ключевого материала в требуемых открытых стандартах, указанных в этой части.

Таблица 15-7 ниже показывает основные примеры командной строки для создания пар Публичного и Частного Ключа, производства сертификатов и цифровой подписи файлов данных.

15-8.5.1 Установка SA

Эта процедура выполняется только один раз. Команда SA-1 в Таблице устанавливает новый набор параметров ECDSA, а команда SA-2 создает "корневой сертификат" SA - их самоподписанный ключ, который удостоверяет их личность.

Когда сервер данных создает запрос подписи сертификата X509 (CSR), SA подписывает его с помощью команды SA-3. Это создает подписанную версию публичного ключа сервера данных. Кодированная PEM версия файла "ds.crt" - это то, что встроено в файлы разрешений и метаданные каталога как "сертификат сервера данных". SA может также выпускать сертификат сервера данных сам для себя.

Таблица 15-8 – Создание пар публичных и частных ключей – базовые команды

Задача	Команда
SA-1 создает частный ключ SA	<code>openssl ecparam -name secp384r1 -genkey -out sapriv.pem</code>
SA-2 создает самоподписанный сертификат SA	<code>openssl req -new -x509 -key sa-priv.pem -sha384 -out sa.crt -days 365</code>
SA-3 создает и подписывает сертификат сервера данных	<code>openssl x509 -req -in ds.csr -CA sa.crt -CAkey sapriv.pem -out ds.crt -sha384 -days 365</code>

15-8.5.2 Настройка сервера данных

Сервер данных настраивает свою личность с помощью SA, используя единовременный процесс, описанный командами DS-1 - DS-5. Таким образом поставляется подписанный SA сертификат серверу данных, который включается в каждую поставку подписанных материалов клиенту данных.

Таблица 15-9 – Команды настройки сервера данных

Task	Command
DS-1 создает частный ключ сервера данных	<code>openssl ecparam -name secp384r1 -genkey -out ds-key.pem</code>

DS-2 выделяет публичный ключ из частного ключа	<code>openssl ec -outform pem -in ds-key.pem -out ds-publickey.pem -pubout</code>
DS-3 создает подписанный запрос	<code>openssl req -new -sha384 -out ds.csr -key ds-key.pem</code>
DS-4 верифицирует полученный от SA сертификат	<code>openssl verify -verbose -CAfile sa.crt ds.crt</code>
DS-5 делает файл данных	<code>echo "hello world" > hw.txt</code>
DS-6 подписывает файл данных	<code>openssl dgst -sha384 -sign ds-key.pem -out signature.bin hw.txt</code>
DS-7 кодирует подпись как Base 64	<code>openssl enc -base64 -in signature.bin -out signature.b64</code>
DS-8 верифицирует подпись	<code>openssl enc -d -base64 -in signature.b64 -out signature.bin</code> <code>openssl dgst -sha384 -verify ds-public-key.pem -signature signature.bin hw.txt</code>

Команды с DS-6 по DS-8 показывают, как может создаваться простой текстовый файл "hello world", подписанный частным ключем сервера данных, для создания подписи ECDSA-P384, а затем верифицироваться. DS-7 создает 64 кодируемую подпись, которая может быть использована для встраивания в файл XML (или в PERMIT.XML или в каталог метаданных, если требуется) согласно соответствующей части S-100.

```
<digitalSignature>
  MGQCMDP17NEJXU7gzwTQAp2lgyDzJd1agCeoz6FZOMGFRmV4sPfzAUh1C3hdj+DF
  3n2n/QIwPYzh15YiBgJ5Aph11kFUjLywzjDZGHYm/GyjxeCL/8FnOviMwccT1xh6
  5fNkL0eg==
</digitalSignature>
```

15-8.6 Пример цифрового сертификата

Цифровые сертификаты PEM кодируются для облегчения обмена и встраивания в XML-файлы. Ниже приведен пример PEM кодирования сертификата сервера данных. Команды, перечисленные в предыдущем разделе, форматируют публичные ключи и запрос на подпись сертификата для связи между SA и DS. При встраивании цифровых сертификатов в элементы XML, строки заголовка и нижнего колонтитула (footer) опускаются.

Файл каталога на основе набора обмена S-100 будет содержать копию всех сертификатов DS, используемых всеми файлами, включенными в набор обмена, за исключением корневого сертификата SA, который устанавливается отдельно конечным пользователем. Идентификатор, представляющий корневой сертификат SA, включается в сертификаты каталога обмена элементом "schemeAdministrator" с атрибутом "id".

Каждый элемент XML, содержащий сертификат, будет иметь уникальный атрибут "id" идентификатора. Каждое определение сертификата XML будет также включать атрибут "issuer", определяющий идентификатор производителя, либо SA

(идентифицируемый идентификатором `schemaAdministrator`), либо координатора домена (сертификат которого также будет включен в набор обмена). В приведенном ниже примере показан фрагмент из заголовка каталога набора обмена с сертификатом SA (не включён) с идентификатором "root". Подписанный SA сертификат сервера данных с идентификатором "DS1" затем включается в PEM закодированный сертификат.

```
<S100XC:certificates>
  <S100CE:schemaAdministrator id="root"/>
  <S100CE:certificate id="DS1" issuer="root">
MIICDjCCAZMCFEvcGmio4FLGYU9VtSiIjkr3n+i6MAoGCCqGSM49BAMDMF0xCzAJ
BgNVBAYTAK1DMRUwEwYDVQQHDAxEZWZhdWx0IENpdHkxHDAaBgNVBAoME0RlZmF1
bHQgQ29tcGFueSBMdGQxCjAIBgNVBAsMAS8xCjAIBgNVBAMMAWQwHhcNMjMxMTMw
MTczOTA0WWhcNMjMxMTMwMTczOTA0WjB7MQswCQYDVQQGEwJNQzEwMBQGA1UECAwN
REFUQVQ9QUk9EVUNFUjEwMC4GA1UECgwnSw50ZXJlYXRpb25hbCBIeWRyb2dyYXBo
aWwMgT3JnYW5pc2F0aW9uMSIwIAYDVQQDDDBlcm46bXJuOmlobzpvcmc6MDBBQTox
ODEwMHYwEAYHKoZIzj0CAQYFK4EEACIDYgAEfnOz0pGcPnvTXIYVhfVwSfM5+gf0
5QRlfcTfswveUijttUhrZJUDZSBf5s15tEEAaseQqDpJJcR9z354GN4uzpqHPELL
zNaahZ+oYBois44W4Y5Qo+NfH5iaRHbmsNOiMAoGCCqGSM49BAMDA2kAMGYCMQDc
lFEyN3iFINm/501mKp/8HwPxnDwkH7tgBnY8PBLQk69vTqP0ow3cieJN44EM9rsC
MQCo+v/K7P1eanGRurkL0stFoEcNySgErIDFQ7sCYF8/E3/onf5/q81wMH66DBJF
IHU=
  </S100CE:certificate>
```

15-8.7 Создание цифровых подписей сервером данных.

Сервер данных создает цифровую подпись для необходимых файлов данных, используя алгоритм ECDSA и их частный ключ, см. раздел 15-8.4.

Все файлы, включенные в набор обмена S-100, должны иметь свои подписи, закодированные как элементы `S100_DatasetDiscoveryMetadata-digitalSignature` или `S100_SupportFileDiscoveryMetadata-digitalSignature`.

Поле `digitalSignatureReference` должно кодироваться как **"ECDSA-384-SHA2"**.

Сертификат сервера данных всегда должен быть снабжен цифровой подписью. Это позволяет OEMу аутентифицировать сертификат, используя публичный ключ SA и проверяя действительность сертификата. Публичный ключ сервера данных можно извлечь из сертификата и использовать для проверки подлинности файла набора данных.

Индивидуальные атрибуты `id` могут быть использованы в качестве поиска OEM, когда цифровая подпись определена. Это уменьшает необходимость повторять сертификат сервера данных каждый раз, когда кодируется подпись.

Те же элементы XML для сертификата сервера данных и цифровой подписи, которые определены в каталоге набора обмена, также используются для цифровой подписи вспомогательных файлов, не включенных в метаданные каталога; например, каталог и файлы разрешений. Они включены в XML-схемы S-100 и содержат все необходимые сертификаты сервера данных, включенные в них.

Поскольку координаторы доменов (например, IMO) могут создавать сертификаты сервера данных для участников своего домена, следующий механизм должен использоваться для обеспечения того, чтобы система клиента данных могла выполнить проверку пути сертификата:

1. Сервер данных всегда должен включать цифровой сертификат своего координатора домена, чтобы убедиться, что OEM клиента данных имеет все сертификаты, необходимые для выполнения полной проверки пути сертификата без какого-либо внешнего доступа.
2. Когда сертификат сервера данных определяется в метаданных каталога, он включает идентификатор сервера данных и ссылку на поставщика. OEM должен искать сертификат поставщика и использовать его для проверки подлинности сервера данных.
3. OEM должен проверить подлинность поставщика сертификата, чтобы проверить правильность сертификата домена, используемого для проверки подлинности сертификата сервера данных. Это делается до проверки подписи в соответствии с ECDSA. Все сертификаты в наборе обмена должны быть аутентифицированы SA, либо напрямую, либо через косвенную аутентификацию одним или несколькими координаторами домена.

Цифровая подпись используется в метаданных каталога (и метаданных вспомогательных файлов) в двух областях:

- Цифровая подпись файла данных ECDSA, пара R, S встраивается в соответствующий элемент XML в соответствии со схемами S-100 XML и закодированным base64; например:

```
<digitalSignature id="sig1" certificateRef="PRIMAR">  
MQQCM DP17NEJXU7gzwTQAp2lgyDzJd1agCeoz6FZOMGFRmV4sPfzAUh1C3hdj+DF3n2n/  
QIwPYzh15YiBgJ5Aph11kFUjLywzjDZGHYm/GyjxeCL/8Fn0viMwccTlxh65fNkL0eg==  
</digitalSignature>
```
- Сертификат сервера данных (который остается неизменным). Это кодируется в соответствии с пунктом 15-8.4 и должно быть встроено в заголовок метаданных каталога. Этот сертификат предоставляет публичный ключ, по которому проверяется цифровая подпись (и содержимое файла). Сам сертификат сервера данных подписывается администратором Схемы (или промежуточным координатором домена) и ответственность за обеспечение того, чтобы отдельно установленный корневой сертификат от SA был доступен в установленной системе. Сертификаты сервера данных должны быть проверены перед проверкой подлинности файла набора данных.

Сертификат сервера данных должен включаться в метаданные набора данных только один раз. Поскольку сертификат не изменяется, на него можно ссылаться по его атрибуту "id", когда на него ссылаются несколько цифровых подписей.

Другая кодировка цифровой подписи – это файл PERMIT.SIGN, который содержит отдельную подпись содержимого файла разрешения, созданного сервером данных, выдающим разрешение. Файл PERMIT.SIGN - это цифровая подпись,

содержащая элементы, определяющие имя файла, цифровую подпись и любые необходимые сертификаты (и промежуточные сертификаты координатора домена).

15-8.8 Дополнительные цифровые подписи.

Дополнительные цифровые подписи могут быть добавлены путем включения дополнительных цифровых подписей в каталог. Это может представлять собой список сертифицированных идентификаторов, подписывающих отдельный ресурс. Это необязательное дополнение к digitalSignature, минимальное являющееся одиночным digitalSignature, проверяющим содержание одного ресурса (например, разрешения, каталога, набора данных или дополнительного файла наборов данных) на соответствие именованному сертификату.

- Дополнительные цифровые подписи имеют собственный тип XML и могут подписывать либо сам ресурс, либо существующую подпись ресурса.
- Дополнительные цифровые подписи ресурса прилагаются к записи в каталоге обмена и имеют тот же формат, что и существующие цифровые подписи. Элемент dataStatus обозначает, относится ли подпись к незашифрованным, сжатым или зашифрованным (и сжатым) ресурсам.
- Цепочки цифровых подписей создаются с использованием атрибута signatureRef. Цепочная digitalSignature подписывает содержимое другой цифровой подписи ресурса в каталоге обмена. В этом случае подписанным содержимым является байт-массив ASN.1, представляющий пару R, S ссылочной подписи.
- Каждая подпись в цепочке требует наличия действительного certificateRef и идентифицирующего атрибута "id".

Эти атрибуты сведены в таблицу 15-10 ниже:

Таблица 15-10 – Дополнительные атрибуты цифровой подписи

Attribute	Purpose
id	Уникальный идентификатор значения цифровой подписи
certificateRef	Публичный ключ, по которому можно проверить подпись. Это возможно только в том случае, если подписанный публичный ключ включен в сам элемент цифровой подписи, в противном случае он является обязательным.
dataStatus	[Только для подписей данных] является ли подпись незашифрованным ресурсом, который только сжат (такой как архив нескольких ресурсов) или зашифрованным (и сжатым)

Полный пример, содержащийся в элементе datasetDiscoveryMetadata, показан ниже. В этом примере метаданные обнаружения наборов данных определили файл данных. Первая подпись "s1" подписывает ресурс набора данных (атрибут "ref" не требуется), подпись "s2" подписывает зашифрованные данные, а подпись "s3" подписывает подпись s2.

```

[datasetDiscoveryMetadata entry]
<S100XC:digitalSignatureValue>
  <S100SE:S100_SE_SignatureOnData id="s1" certificateRef="PROD1"
    dataStatus="unencrypted">(sig. omitted)</S100SE:S100_SE_SignatureOnData>
</S100XC:digitalSignatureValue>

<S100XC:digitalSignatureValue>
  <S100SE:S100_SE_SignatureOnData id="s2" certificateRef="RENC1"
    dataStatus="encrypted">(sig. omitted)</S100SE:S100_SE_SignatureOnData>
</S100XC:digitalSignatureValue>

<S100XC:digitalSignatureValue>
  <S100SE:S100_SE_SignatureOnSignature id="s3" certificateRef="DIST1"
    signatureRef="s2">(sig. omitted)</S100SE:S100_SE_SignatureOnSignature>
</S100XC:digitalSignatureValue>

```

15-8.9 Проверка целостности и цифровой идентичности данных с цифровой подписью S-100

Проверка цифровой подписи является алгоритмом, который работает на трех независимых частях данных (все отформатированы в соответствии с этой частью S-100):

1. **Содержимое**, требующее проверки (формат содержимого является произвольным);
2. **Публичный ключ**, соответствующим образом кодированный. В принятом алгоритме ECDSA этот публичный ключ является одним числом вместе с набором параметров ECDSA (три числа);
3. **Подпись**. В алгоритме ECDSA подпись состоит из двух чисел, по соглашению они называются R и S (пара R, S).

Процесс проверки подписи определяет, аутентифицирует ли пара R, S содержимое по данному публичному ключу. Это может привести только к результату true или false.

Проверка цифровой подписи ECDSA дает два результата:

- **Аутентификация**: Система проверяет публичный ключ сервера данных ("content") и подпись в сертификате сервера данных ("signature") с помощью публичного ключа SA ("Public Key"), чтобы подтвердить, что публичный ключ поставщика в сертификате действителен и что сервер данных является добросовестным членом схемы защиты данных S-100. Если имеется координатор домена, также должна проверяться подлинность координатора домена с помощью публичного ключа SA.
- **Проверка целостности**: Система проверяет подпись файла данных ("signature") и публичный ключ сервера данных в сертификате сервера данных ("Public Key") в файле данных ("content"). Это проверяет содержимое файла данных.

Если проверка прошла успешно, то она доказывает, что данные не были повреждены каким-либо образом и что личность сервера данных в подписях набора

данных проверяется удостоверением SA, как определено в корневом сертификате SA. Корневой сертификат SA, содержащий публичный ключ, должен быть установлен отдельно в системе конечного пользователя и не упакован с метаданными набора данных.

15-8.10 Спецификации MRN

Для поддержки обнаруживаемости ресурсов части 17 Набор обмена, следующие пространства имен MRN определяются этой частью S-100. Они предназначены для использования в целях обнаружения дополнительных ресурсов набора данных с помощью уникального криптографического хеша или цифровой подписи. Алгоритм, используемый для определения хеша или подписи, встроен в MRN.

Таблицы 15-11 и 15-12 ниже показывают спецификации цифровой подписи и хеш-MRN в S-100. Все поля обязательны и не зависят от регистра.

Таблица 15-11 – Цифровая подпись MRN S-100

Имя	Значение	Пример
префикс	<code>urn:mrn:iho:s100:dsig</code>	
алгоритм	Из digitalSignatureReference (Part XX 4a-5)	<code>ECDSA-384-SHA2</code>
значение	Расчетное значение цифровой подписи	<code>MGUCMQCd9T4ggpAeVA/6zB0HWCXTsUOaD56lM4UitkNXrYa5rURtLwiWH2D/ZkmYRY1LTO8CMHIYHpBXvr7HwY6+W36bXnR5ylc8QTN7vc9WH/Zmo5Ck1IH02RUBS286RnYXUEP3WQ==</code>
пример	<code>urn:mrn:iho:s100:dsig:ecdsa:MGUCMQCd9T4ggpAeVA/6zB0HWCXTsUOaD56lM4UitkNXrYa5rURtLwiWH2D/ZkmYRY1LTO8CMHIYHpBXvr7HwY6+W36bXnR5ylc8QTN7vc9WH/Zmo5Ck1IH02RUBS286RnYXUEP3WQ==</code>	

Таблица 15-12 – Криптографический хеш MRN S-100

Имя	Значение	Пример
префикс	<code>urn:mrn:iho:s100:hash</code>	
алгоритм	digitalSignatureReference (Part XX 4a-4.5)	<code>SHA-256</code>
значение	Вычисленный криптографический хеш выраженный как шестнадцатеричный	<code>a948904f2f0f479b8f8197694b30184b0d2ed1c1cd2a1ec0fb85d299a192a447</code>
пример	<code>urn:mrn:iho:s100:hash:sha256:a948904f2f0f479b8f8197694b30184b0d2ed1c1cd2a1ec0fb85d299a192a447</code>	

15-8.11 Метаданные каталога обмена и спецификация одиночного элемента схемы

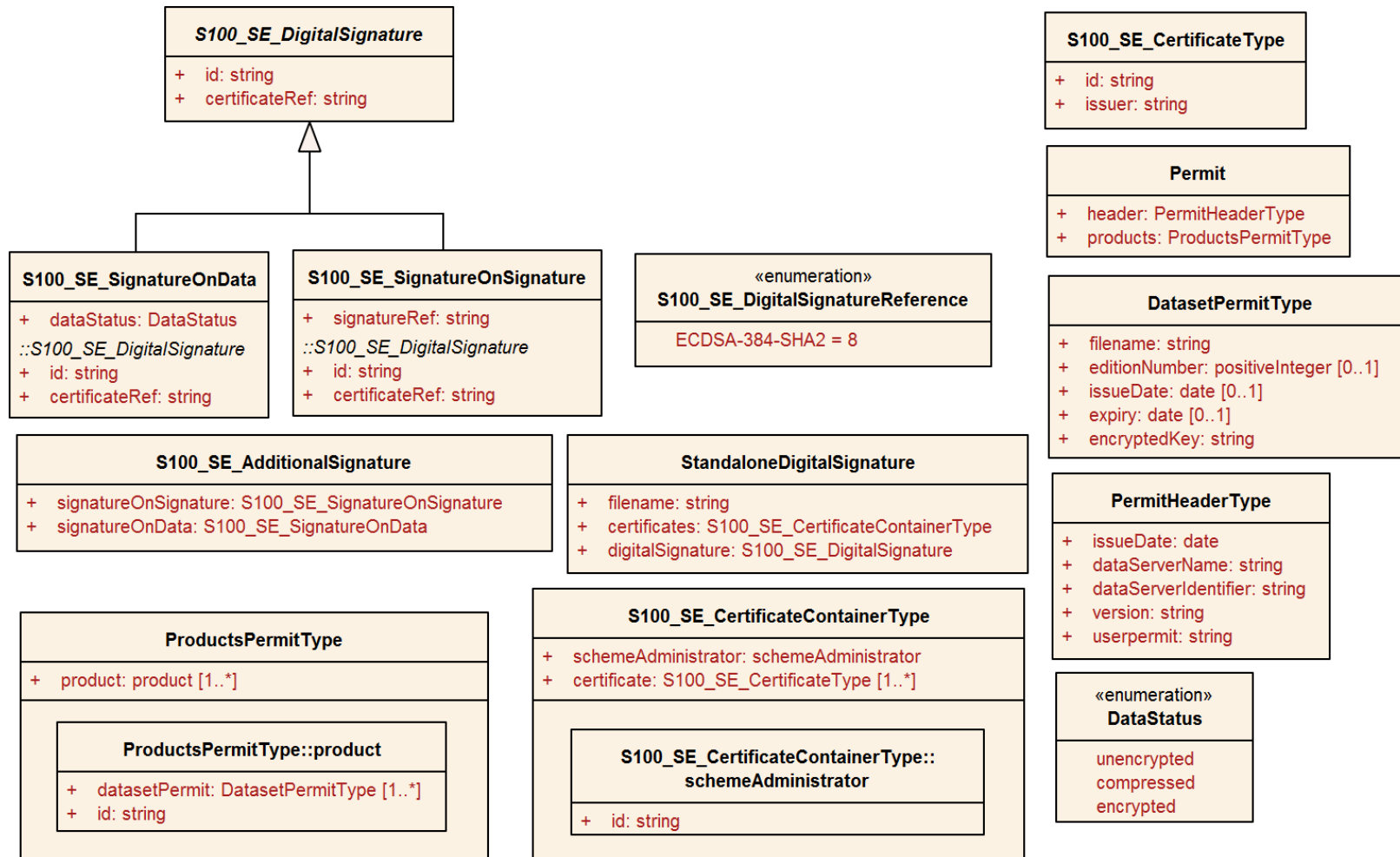


Рисунок 15-8 – Защита данных – детали класса

15-8.11.1 S100_SE_CertificateContainerType

Имя роли	Имя	Описание	Мнж	Тип данных	Примечания
Класс	S100_SE_CertificateContainerType	Набор подписанных сертификатов публичного ключа	-	-	Используется в каталогах обмена S-100 Часть 17
Атрибут	schemeAdministrator	Удостоверение администратора схемы	0..1	CharacterString	Подлинность администратора схемы содержится в атрибуте "id" элемента schemeAdministrator. Сертификат администратора схемы <u>НЕ</u> включается в метаданные каталога, так как он проверяется системой независимо. Кодирование ИНО как schemeAdministrator следующее: <S100SE:schemeAdministrator id="ИНО"/>
Атрибут	certificate	Подписанный сертификат публичного ключа	1..*	Представленная в кодировке Base64 Character String	Соответствует кодированию X.509. Содержит идентификатор с цифровой подписью.

15-8.11.2 StandaloneDigitalSignature

Имя роли	Имя	Описание	Мнж	Тип данных	Примечания
Класс	StandaloneDigitalSignature	Одиночная цифровая подпись	-	-	
Атрибут	filename	Имя файла подписанного контента	1	CharacterString	Имя файла подписанного ресурса
Атрибут	certificates	Любые сертификаты, необходимые для аутентификации подписи администратора схемы	1	S100_SE_CertificateContainerType	
Атрибут	signature	Одиночная цифровая подпись	1	S100_SE_DigitalSignature	Подпись ресурса файла

15-8.11.3 S100_SE_DigitalSignature

Класс S100_SE_DigitalSignatureValue реализуется или как S100_SE_SignatureOnData (цифровая подпись определенного ресурса) или как дополнительная цифровая подпись, определенная с использованием класса S100_SE_AdditionalSignature, каждая из которых или элемент S100_SE_SignatureOnData, или элемент S100_SE_SignatureOnSignature, как это описано в разделе 15-8.8. Таким образом метаданные S-100 Части 17 позволяют иметь несколько цифровых подписей, одна обязательная S100_SE_SignatureOnData и любое количество дополнительных подписей, или данных или других подписей.

15-8.11.4 S100_SE_SignatureOnData

Имя роли	Имя	Описание	Мнж	Тип данных	Примечания
Класс	S100_SE_SignatureOnData		-	Кодированное по Base64 значение цифровой подписи (раздел 15-8.4)	-
Атрибут	id	Идентификатор цифровой подписи	1	CharacterString	Каждая запись подписи имеет уникальный идентификатор
Атрибут	certificateRef	Подписанный публичный ключ	1	CharacterString	Идентификатор сертификата, в отношении которого удостоверяется цифровая подпись
Атрибут	dataStatus	Цифровая подпись	1	DataStatus	Значение цифровой подписи, вычисленное по определенному алгоритму

15-8.11.5 S100_SE_SignatureOnSignature

Имя роли	Имя	Описание	Мнж	Тип данных	Примечания
Класс	S100_SE_SignatureOnSignature		-	Кодированное по Base64 значение цифровой подписи (раздел 15-8.4)	-
Атрибут	id	Идентификатор цифровой подписи	1	CharacterString	Каждая запись подписи имеет уникальный идентификатор

Атрибут	certificateRef	Подписанный публичный ключ	1	CharacterString	Идентификатор сертификата, в отношении которого удостоверяется цифровая подпись
Атрибут	signatureref	Ссылочная цифровая подпись	1		

15-8.11.6 DataStatus

Имя роли	Имя	Описание	Примечания
Перечень	DataStatus	Состояние данных при создании цифровой подписи	
Атрибут	Unencrypted	Данные не зашифрованы и не сжаты	Например, поддерживающие ресурсы
Атрибут	Encrypted	Данные сжаты и зашифрованы	Например, защищенные от копирования наборы данных
Атрибут	Compressed.	Данные только сжаты	Например, архивы нескольких ресурсов

15-8.11.7 S100_SE_DigitalSignatureReference

Имя роли	Имя	Описание	Код	Примечания
Перечень	S100_SE_DigitalSignatureReference	Ссылка на криптографический алгоритм, используемый в реализации Части 15.		В настоящее время только ECDSA используется в реализации S-100 для передачи файловых данных в ECDIS. Другие значения включены для совместимости с другими системами, созданными по другим стандартам. См. раздел 15-8.4
Значение	RSA		1	RSA с ключом длиной ≥ 2048 бит
Значение	DSA		2	DSA с ключом длиной ≥ 2048 бит
Значение	ECDSA		3	ECDSA с ключом длиной ≥ 224 бит.
Значение	ECDSA-224-SHA2-224		4	224 бит ECDSA с SHA2-224 хешированием
Значение	ECDSA-224-SHA3-224		5	224 бит ECDSA с SHA3-224 хешированием
Значение	ECDSA-256-SHA2-256		6	256 бит ECDSA: SHA2-256

Значение	ECDSA-256-SHA3-256		7	256 бит ECDSA: SHA3-256
Значение	ECDSA-384-SHA2		8	384 бит ECDSA: SHA2-384
Значение	ECDSA-384-SHA3		9	384 бит ECDSA: SHA3-384
Значение	AES-128		10	ключ AES 128 бит
Значение	AES-192		11	ключ AES 192 бит
Значение	AES-256		12	ключ AES 256 бит

15-9 Глоссарий схемы защиты данных S-100 и компьютерные термины

Список сокращений смотрите S-100, часть 0, раздел 0-2. Список терминов и определений смотрите в S-100 приложение А.

Таблица 15-13 – Термины схемы защиты данных S-100

AES	Advanced Encryption Standard, алгоритм шифрования, используемый в схеме
Data Permit	Файл, содержащий зашифрованные ключи продукта, необходимые для расшифровки лицензированных продуктов. Он создается специально для конкретного пользователя.
Data Client	Термин, используемый для обозначения конечного пользователя, получающего зашифрованную информацию ЭНК. Клиент данных будет использовать программное обеспечение (например, СОЭНКИ) для выполнения многих операций, подробно описанных в схеме. Как правило, пользователь СОЭНКИ.
Data Server	Термин, используемый для обозначения организации, производящей зашифрованные файлы данных или выдающей разрешения на использование набора данных конечным пользователям.
M_ID	Уникальный идентификатор, присвоенный SA каждому производителю. Серверы данных используют его, чтобы определить, какой ключ M_KEY использовать при расшифровке Userpermit
M_KEY	Уникальный идентификационный ключ изготовителя СОЭНКИ, предоставленный администратором схемы OEMy. Он используется OEM для шифрования HW_ID при создании userpermit
HW_ID	Уникальный идентификатор, присвоенный OEMом каждой реализации своих систем. Это значение зашифровано с помощью уникального M_KEY OEMом и поставляется клиенту данных как userpermit. Этот метод позволяет клиентам данных приобретать лицензии для расшифровки наборов данных ЭНК.
PKCS	Public Key Cryptography Standards (Стандарты Криптографии Публичного Ключа)
IV	Initialization Vector (вектор инициализации, используемый алгоритмом шифрования AES-CBC)
SA	Scheme Administrator. (Администратор схемы). Ответственный в ИНО за поддержание и координацию всех оперативных аспектов и документации системы защиты
SHA	Secure Hash Algorithm
SSK	Self Signed Key (Self Signed Certificate File)
User Permit	Зашифрованная форма HW-ID уникально идентифицирующая систему клиента данных.
Таблица 15-10 – Компьютерные термины	
CRC	Cyclic Redundancy Check
XOR	Exclusive OR