Операционная система "Магистра" 1.3

Руководство программиста - пользователя

Платформа ST23YL18

(сокращенная версия)

Содержание

1 ВВЕДЕНИЕ 5
2 ЖИЗНЕННЫЙ ЦИКЛ КАРТЫ 5
3 ПРОТОКОЛЫ ВЗАИМОДЕЙСТВИЯ ТЕРМИНАЛЬНОГО ОБОРУДОВАНИЯ С КАРТОЙ 6
3.1 Протокол "T=0" (ISO7816) 7
4 КОНТРОЛЬ И ДИАГНОСТИКА КАРТЫ 8
5 ФАЙЛОВАЯ СИСТЕМА 9
5.1 Файлы директорий 10
5.2 Файлы данных 11
5.2.1 Бинарные файлы (BF) 11
5.2.2 Файлы записей (FRF, CRF, VRF) 11
5.2.3 Файлы BER-TLV записей (TF) 12
5.3 Специальные файлы 12
5.3.1 Файлы ключей KF 13
5.3.2 Файлы правил разграничения доступа 14
5.3.3 Файлы хранения SE 15
6 СИСТЕМА РАЗГРАНИЧЕНИЯ ДОСТУПА 15
6.1 Санкции 15
6.2 Атрибуты доступа 16
6.3 Правила доступа 18
6.4 Авторизация доступа 19
7 КРИПТОГРАФИЧЕСКИЕ ОПЕРАЦИИ И ПРОТОКОЛЫ 20
7.1 Secure Environment 20
7.2 Аутентификация 20
7.2.1 Аутентификация по паролю 21
7.2.2 Внешняя аутентификация 21
7.2.3 Внутренняя аутентификация 22
7.2.4 Взаимная аутентификация по симметричной схеме 23
7.2.5 Взаимная аутентификация по несимметричной схеме 23
7.3 Защищенный обмен сообщениями 23
7.4 Криптографические сервисы 24
8 ПРИЛОЖЕНИЯ 24
9 КОМАНДЫ ОС 25
9.1 Команды управления файлами 25
9.1.1 CREATE FILE 26
9.1.2 ACTIVATE FILE 27
9.1.3 DEACTIVATE 28
9.1.4 DELETE FILE 29
9.1.5 TERMINATE CARD USAGE 30
9.2 Команды навигации 31
9.2.1 SELECT 31
9.3 Команды работы с данными BER-TLV 33
9.3.1 GET DATA 34
9.3.2 PUT DATA 36
9.4 Команды работы с бинарными файлами 37

9.4.1 READ BINARY ... 37
9.4.2 UPDATE BINARY ... 38
9.4.3 WRITE BINARY ... 39
9.5 КОМАНДЫ РАБОТЫ С ФАЙЛАМИ ЗАПИСЕЙ ... 40
9.5.1 APPEND RECORD ... 42
9.5.2 READ RECORD ... 42
9.5.3 UPDATE RECORD ... 43
9.6 КОМАНДЫ РАБОТЫ С ПРАВИЛАМИ ДОСТУПА ... 44
9.6.1 GET DATA ... 44
9.6.2 PUT DATA ... 45
9.7 КРИПТОГРАФИЧЕСКИЕ КОМАНДЫ ... 46
9.7.1 MANAGE SECURITY ENVIRONMENT ... 46
9.7.2 GENERATE KEY PAIR ... 47
9.7.3 PERFORM SECURITY OPERATION ... 48
9.8 КОМАНДЫ АУТЕНТИФИКАЦИИ ... 55
9.8.1 GET CHALLENGE ... 55
9.8.2 INTERNAL AUTHENTICATE ... 56
9.8.3 EXTERNAL AUTHENTICATE ... 57
9.8.4 GENERAL AUTHENTICATE ... 58
Взаимная аутентификация по симметричной схеме ... 58
Взаимная аутентификация по несимметричной схеме. ... 59
9.8.5 VERIFY ... 60
9.8.6 CHANGE REFERENCE DATA ... 61
9.8.7 RESET RETRY COUNTER ... 62
9.9 СЕРВИСНЫЕ КОМАНДЫ ... 63
9.9.1 VALIDATE CARD ... 63
9.9.2 HANG CARD ... 65
9.10 СПРАВОЧНИК ПО КОМАНДАМ ... 66
9.11 СПРАВОЧНИК ПО СТАТУСАМ КОМАНД ... 67
10 СТРУКТУРЫ ДАННЫХ НА КАРТЕ ... 70
10.1 FCI, FCP, FMD ... 70
10.2 ФАЗЫ ЖИЗНИ ФАЙЛА ... 72
10.3 ТИПЫ ФАЙЛОВ ... 72
10.4 СПОСОБ КОДИРОВАНИЯ ДАННЫХ ... 72

Сокращения

AID	Application Identifier – имя приложения
ARF	файл правил доступа
ATR	Answer To Reset – ответ карты на сигнал сброса
BER-TLV	формат представления записей переменной длины с тэгом
BF	неструктурированный файл данных
CRF	циклический файл записей фиксированной длины
CRT	SE Control Reference Template – шаблон настроек среды безопасности
DF	файл-директория
DS	Digital Signature – см. ЭЦП
EEPROM	энергонезависимая память карты
FCI	File Control Information
FCP	File Control Parameters
FMD	File Management Data
FRF	файл записей фиксированной длины
KF	файл ключа
MF	корневая директория файловой системы
MSE	команда MANAGE SECURITY ENVIRONMENT
OTP	область одноразовой записи в EEPROM (One-Time Programmable)
PSO	команда PERFORM SECURITY OPERATION
RFU	Reserved for Future Use – зарезервировано
SE	Security Environment – среда безопасности
SEF	файл для хранения SE
TF	файл объектов данных в формате BER-TLV
TLV	Tag-Length-Value – способ хранения и передачи данных
VRF	файл записей переменной длины
ОС	операционная система
ФЖ	фаза жизни
ФКН	приложение Функциональный Ключевой Носитель
ЭЦП	электронная цифровая подпись

1 ВВЕДЕНИЕ

Данный документ адресован разработчикам, планирующим использовать смарт-карты в своих системах и самостоятельно разрабатывать соответствующие приложения для смарт-карт.

В данном документе описывается версия ОС Магистра, реализующая стандарты ISO 7816-4,8,9 и предназначенная, в первую очередь, для реализации на ее основе идентификационных и криптографических приложений.

Отличительными особенностями ОС являются:

- поддержка Российской криптографии
- поддержка контактного интерфейса ISO 7816 (протокол T0)
- поддержка файловой системы на основе группы стандартов ISO 7816
- поддержка механизма расширений.

ОС реализована на микроконтроллере ST23YL18 производства компании STMicroelectronics.

Приложение смарт-карты разрабатывается как совокупность структурных элементов (файлов, директорий) для хранения разнообразных данных, доступ к которым (запись, чтение, использование) определяется самим разработчиком.

2 ЖИЗНЕННЫЙ ЦИКЛ КАРТЫ

Жизненный цикл карты состоит из 4-х основных фаз:

- Фаза инициализации
- Фаза эксплуатации
- Фаза блокирования

Переход между фазами жизни карты происходит последовательно и необратимо.

Фаза инициализации применяется для первоначального форматирования карты и загрузки расширений. Фаза инициализации характерна отсутствием файловой системы. Фаза инициализации завершается переводом карты на основной режим функционирования - на фазу эксплуатации.

Инициализация карты включает следующие этапы:

- форматирование карты,
- загрузка расширений,
- создание файловой системы.

На этой фазе действует специальный набор команд инициализации. В результате успешного создания файловой системы карта переходит на фазу эксплуатации.

Фаза эксплуатации предназначена для практического использования карты. На этой фазе можно задействовать все функциональные ресурсы карты, в энергонезависимой памяти существует файловая система, которая подчиняется правилам жизненного цикла, определенным в ISO 7816-9 (у каждого файла и директории есть свой жизненный цикл).

Фаза блокирования карты предназначена для прекращения функционирования карты по причине внутренних отказов или по инициативе терминального оборудования. Фаза блокировки гарантирует невозможность чтения и модификации информации на карте и эквивалентна физическому выведению карты из строя.

На фазе эксплуатации каждый файл и директория имеют свой жизненный цикл,

который состоит из следующих фаз жизни:

- Инициализация
- Использование
- Блокирование (обратимое или необратимое)

Переходы между фазами жизненного цикла файла

По умолчанию файл создается с фазой жизни «инициализация», но в команде создания можно указать фазу жизни файла «использование», тогда «инициализация» будет пропущена.

Фаза «использование» – основная фаза жизни файла, на которой действуют все правила разграничения доступа в полном объеме.

Фаза «инициализация» отличается тем, что перечисленные ниже виды доступа считаются безусловно открытыми:

- добавить подчиненный файл к DF;
- добавить данные к любому файлу;
- перевести файл на фазу использования.

Когда файл находится на фазе обратимого блокирования, к нему применимы только следующие операции, защищенные соответствующими атрибутами доступа:

- удалить файл;
- активизировать файл.

В блокированный DF нельзя войти, т.е. выбрать вложенный в него файл. Однако можно выбрать сам блокированный DF, чтобы активизировать его или удалить вместе со всеми вложенными файлами.

3 ПРОТОКОЛЫ ВЗАИМОДЕЙСТВИЯ ТЕРМИНАЛЬНОГО ОБОРУДОВАНИЯ С КАРТОЙ

Карта ST23YL18 поддерживает контактный интерфейс и, соответственно, следующий протокол обмена с терминалом:

- Протокол ***ISO 7816 T=0***.

Сразу после старта карта выполняет быстрое самотестирование, анализирует

содержимое буфера транзакций и, при необходимости, восстанавливает целостность EEPROM после прерванной транзакции. После этого карта отвечает терминалу последовательностью данных ATR.

Восстановление целостности EEPROM может оказаться достаточно длительной процедурой. Если терминал снял питание, не дождавшись ответа карты, то в этом случае восстановление данных будет продолжено при следующем рестарте. Таким образом, максимум за несколько итераций целостность данных будет восстановлена.

3.1 Протокол "T=0" (ISO7816)

Карта использует прямой порядок кодирования байтов (direct convention, см. ISO 7816-3, раздел 6.4).

Поддерживается процедура PPS и выбор скорости обмена до 16 тактов/бит, что примерно равно 223200 бит/сек при стандартной частоте тактирования 3.57 МГц.

Размер передаваемых данных ограничен размером **255** байт в запросе и **256** байт в ответе.

Протокольные байты ATR

Байт	значение	пояснение
TS	3B	direct convention
T0	93	8 – признаки присутствия TD1 3 - число байтов истории
TA1(*)	96	максимальная частота 5MHz минимальный период передачи бита ETU = F/D тактов где F=512, D=32 (т.е. ETU >= 15 тактов/бит)
TD1	00	предлагается протокол T0 нет дальнейших байтов TA2..TD2

(*) параметр может быть скорректирован при инициализации карты

Байты истории на фазе инициализации

Байт	значение	пояснение
HB1	80	category indicator byte в конце признак фазы жизни в формате CompactTLV
HB2..3	81 03	фаза жизни карты – инициализация

Стандартные байты истории на фазе эксплуатации

Байт	значение	пояснение
HB1	80	category indicator byte в конце признак фазы жизни в формате CompactTLV
HB2..3	31 C0	Card service data byte - разрешен выбор приложения по полному и частичному имени
HB4..6	72 F7 41	Card capabilities - доступен выбор DF по полному и частичному имени, по пути и по идентификатору - поддерживаются короткие идентификаторы EF - поддерживаются номера записей - поддерживаются идентификаторы (тэги) записей - WRITE BINARY выполняет OR - единицей адресации в BF является 1 байт
HB7..8	81 07	фаза жизни карты – эксплуатация

По договоренности с заказчиком карты, поставщик может изменить протокольные байты ATR, в том числе указав значение байта TA1 и установив, таким образом, скорость по-умолчанию для взаимодействия с терминальным устройством.

Если в контексте MF существует запись с тэгом 5F52 и размером от 1 до 15 байт, то ее содержимое будет выдаваться в качестве байтов истории вместо стандартных (значение T0 изменится на 8X, где X – число байтов истории).

На фазе блокирования карта не выдает ATR.

Байты истории могут быть получены командой GET_DATA (при этом MF не обязательно должен быть текущим файлом) с параметрами:

- ***5F51*** – ATR полностью
- ***5F52*** – байты истории

4 КОНТРОЛЬ И ДИАГНОСТИКА КАРТЫ

В ОС карты предусмотрены ряд автоматических (не отключаемых) и дополнительных (запускаемых вручную) средств контроля исправности карты и целостности данных.

К автоматическим средствам контроля относятся:

- Самотестирование карты при старте.
- Аппаратный механизм коррекции однократных ошибок в EEPROM и детектирования многократных ошибок.
- Программно-вычисляемая контрольная сумма заголовков файлов, содержимого служебных файлов и пользовательских данных[1].
- Механизм буферизации записи и поддержки транзакций.

При невозможности восстановить целостность аппаратно-программной среды исполнения, карта переходит в состояние "MUTE" ("повисает"). В этом случае восстановление работы карты возможно только после подачи сигнала «Reset». Причинами «повисания» могут быть:

[1] включается пользователем при создании файла

- Ошибка выполнения самотестирования[2]
- Выход параметров, контролируемых датчиками, из диапазона допустимых значений.
- Обнаружение невосстанавливаемой ошибки в EEPROM.
- Невозможность произвести безошибочную запись в EEPROM.

К дополнительным средствам контроля и диагностики относятся:

- Диагностические флаги.
- Команда диагностики VALIDATE CARD (п.9.10.1), которая позволяет:
  - проверить диагностические флаги;
  - проверить весь EEPROM на предмет наличия неисправных блоков;
  - выполняет диагностику оборудования;
  - вычислить некриптографическую контрольную сумму двоичного кода ОС;
  - вычислить криптографическую контрольную сумму двоичного кода ОС.

В ОС определены следующие диагностические флаги:

- Флаг наличия неисправленной ошибки чтения в области файловой системы EEPROM
- Флаг наличия неисправленной ошибки чтения в области буфера транзакций
- Флаг наличия неисправленной ошибки чтения в служебных областях EEPROM или в области кода расширения
- Флаг выявления ошибки записи в области файловой системы EEPROM
- Флаг выявления ошибки записи в области буфера транзакций
- Флаг выявления ошибки записи в служебных областях EEPROM или в области кода расширения

При наличии специальных требований заказчика, карта может выполнять расширенные процедуры самотестирования. Вследствие их значительной продолжительности, эти процедуры выполняются после выдачи ATR, но до выполнения картой первой команды от терминального оборудования. В случае выявления этими процедурами ошибок, карта выдаст соответствующий код ошибки в ответ на первую команду, вместо ее выполнения. Из всего множества команд будет разрешено выполнение лишь только команды диагностики[3].

К расширенным процедурам диагностики относятся:

- проверка некриптографической контрольной суммы ОС
- статистическая проверка ДСЧ
- проверка криптографического сопроцессора

5 ФАЙЛОВАЯ СИСТЕМА

Файловая система карты состоит из иерархически организованных файлов. Максимальный уровень вложенности директорий не ограничен.

В ОС реализовано несколько типов файлов, которые можно разделить на следующие категории:

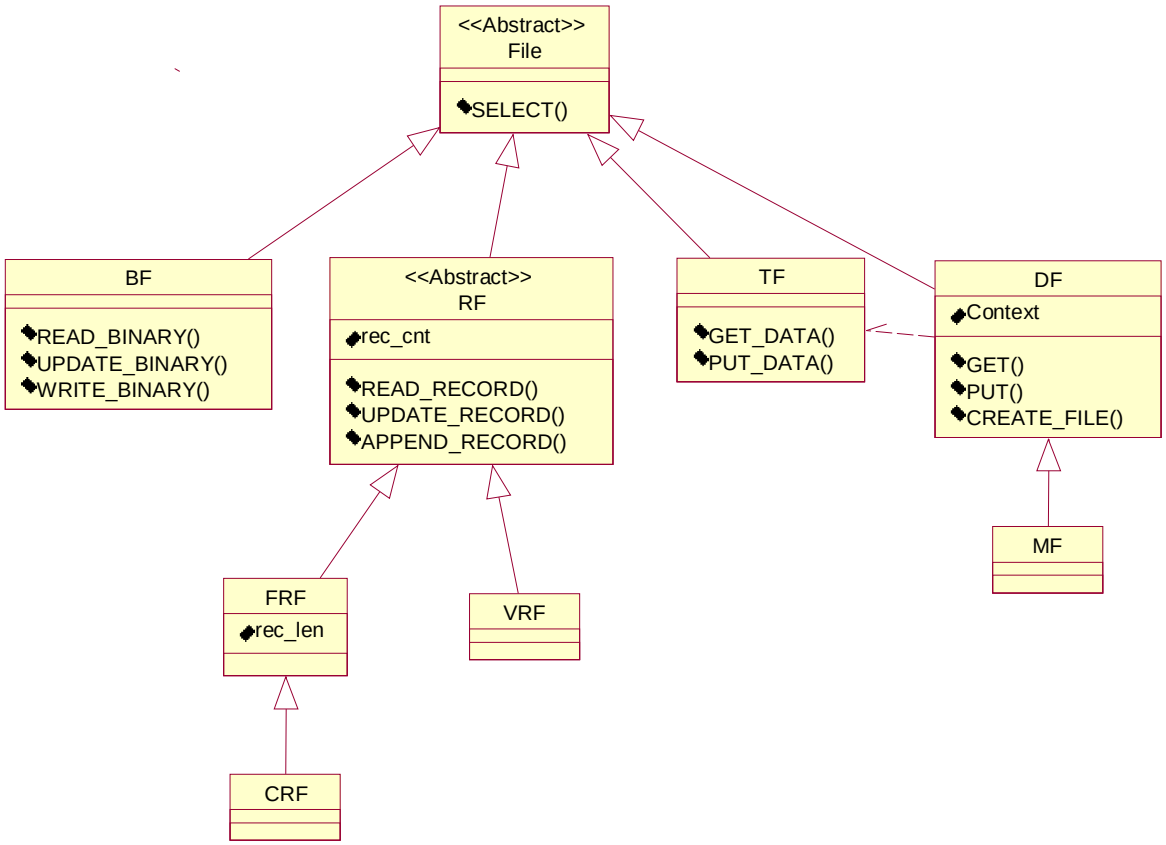
- Файлы директорий:
  - MF корневой файл
  - DF любая директория

[2] в контактном режиме перед «зависанием» может быть выдан аварийный ATR

[3] при необходимости можно запретить даже обработку команды диагностики (“MUTE”)

- Файлы данных общего назначения:
  - BF бинарный файл без внутренней структуры
  - FRF файл записей фиксированной длины
  - CRF циклический файл записей фиксированной длины
  - VRF файл записей переменной длины (в формате Simple-TLV)
  - TF файл записей в формате BER-TLV
- Специальные файлы:
  - KF ключи и пароли
  - ARF правила разграничения доступа
  - SEF файл для хранения SE (настройки среды защиты)

Иерархия типов файлов



Команды управления жизненным циклом файлов см. в разделе «Команды управления файлами» (п.9.1).

Команды навигации по файловой системе см. в разделе «Команды навигации» (п.9.2).

Атрибуты доступа для файлов см. в разделе «Атрибуты доступа» (п. 6.2).

Далее различные типы файлов рассматриваются подробнее.

5.1 Файлы директорий

Файлы директорий предназначены для выстраивания иерархических отношений в файловой системе. Корневой файл называется MF, файлы директорий – DF. MF является разновидностью DF.

Можно установить фиксированный размер DF, тогда ОС гарантирует, что DF вместе со всеми вложенными файлами не превысит заданного размера и, с другой стороны, указанный объем памяти резервируется за данным DF. Если размер DF не указан, то он

может занять все свободное пространство в объемлющем DF.

У каждого DF может быть контекст – область для хранения объектов данных, представленных в формате BER-TLV, ассоциированных с данным DF. Для работы с контекстом используются команды GET DATA и PUT DATA, как для работы с файлом TF. Чтение контекста всегда разрешено. Запись контекста также всегда разрешена на фазе жизни «инициализация».

AID (имя DF) является одним из полей контекста и должен находиться в объекте данных с тэгом ***4F***. Его можно записать как при создании файла, так и после, командой PUT DATA.

Весь контекст выдается наружу в составе FMD (п.10.1).

Основные операции над DF:

Операция	Команда	Право доступа
добавление вложенного файла	CREATE FILE	Create Child File свободно на ФЖ Init
чтение контекста	GET DATA	всегда свободно
запись в контекст	PUT DATA	Put Context; свободно на ФЖ Init

5.2 Файлы данных

Файлы данных предназначены для хранения пользовательских данных, которые никак не интерпретируются ОС.

5.2.1 Бинарные файлы (BF)

Бинарные файлы предназначены для хранения неструктурированных данных.

Максимальный размер файла, доступный для чтения и записи в командах ISO 7816, составляет 0x80FE (т.к. максимальное адресуемое смещение 0x7FFF).

Основные операции над BF:

Операция	Команда	Право доступа
чтение данных	READ BINARY	Read
обновление данных	UPDATE BINARY	Update
наложение новых данных на старые операцией OR	WRITE BINARY	Write

Детальное описание команд см. в разделе "Команды работы с бинарными файлами" (п.9.4).

5.2.2 Файлы записей (FRF, CRF, VRF)

В файлах записей фиксированной длины (FRF) адресация к записям производится по номеру, причем в CRF первой считается самая свежая запись. Максимальный размер записи 255 байт.

В файлах записей переменной длины (VRF) записи хранятся в формате SimpleTLV с 1-байтным тэгом. К записям можно адресоваться как по номеру, так и по тэгу. В файле

может быть несколько записей с одинаковым тэгом. Максимальный размер тела записи (без тэга и поля длины) 254 байта.

Во всех файлах записей определен указатель на текущую запись, при помощи которого можно последовательно двигаться по записям. При выборе файла он не установлен. В файлах VRF можно использовать тэг записи в качестве фильтра, тогда указатель на текущую запись будет «останавливаться» только на записях с указанным тэгом, игнорируя остальные.

В файлах VRF обновление записи возможно только при совпадении длин старой и новой записи.

Любой файл записей может содержать не больше 255 записей.

Основные операции над файлами xRF:

Операция	Команда	Право доступа
Добавить запись	APPEND RECORD	Append (=Update для CRF) свободно на ФЖ Init (для CRF свободно только до момента полного заполнения файла)
Обновить запись	UPDATE RECORD	Update
Прочитать запись	READ RECORD	Read

Детальное описание команд см. в разделе «Команды работы с файлами записей» (п.9.5).

5.2.3 Файлы BER-TLV записей (TF)

Файлы TF содержат объекты данных в формате BER-TLV.

Максимальный размер файла ограничен только размером файловой системы. Максимальный размер объекта **255** байт[4]. Допускаются тэги размером до 3 байтов.

Обновление записи возможно только при совпадении длин старой и новой записи.

Основные операции над TF:

Операция	Команда	Право доступа
Запись данных	PUT DATA	Put свободно на ФЖ Init в случае добавления данных с новыми тэгами
Чтение данных	GET DATA	Get

Детальное описание команд см. в разделе «Команды работы с данными BER-TLV» (п.9.3).

5.3 Специальные файлы

Специальные файлы необходимы для функционирования ОС. Их структура и содержание строго регламентировано ОС.

[4] На платформе ST23YL18 генерация ключа RSA не поддерживается

5.3.1 Файлы ключей KF

Файлы ключей предназначены для хранения любых видов симметричных, асимметричных ключей и паролей. В файле ключа хранится только один ключ или пароль.

Симметричные ключи и пароли всегда загружаются снаружи. Асимметричные ключи могут быть сгенерированы внутри карты или загружены снаружи[5]. Открытая часть асимметричного ключа может быть выдана наружу. Файл с асимметричным ключом может содержать либо ключевую пару, либо только открытую часть ключа (например, открытый ключ центра сертификации).

Каждый KF обязан иметь уникальный идентификатор, который является 1-байтовой целой величиной в диапазоне от 1 до 127. Если ключ используется для внешней аутентификации, то идентификатор ключа выступает в качестве идентификатора санкции.

Ключ не может быть использован непосредственно, если у него установлен хотя бы один из признаков назначения ключа:

- требуется диверсификация ключа;
- требуется генерация сессионного ключа.

В этом случае необходимо сгенерировать производный ключ, который наследует от мастер-ключа все остальные атрибуты.

Загрузка ключа из сертификата рассматривается как диверсификация (в логическом смысле) и требует установки соответствующего бита при создании KF.

Для ключей ГОСТ Р 34.10-2001 может быть определен режим маскирования ключа для обеспечения большей безопасности. В этом случае ключ хранится в маскированной форме, каждое его применение автоматически вызывает его перемаскирование.

Для ключей RSA открытй ключ состоит из пары (e, n) – открытой экспоненты (8 байт) и модуля (128 байт). Секретный ключ содержит только секретную экспоненту d (128 байт).

Основные операции над ключом:

Операция	Команда	Право доступа
Первичная загрузка	CHANGE REFERENCE DATA	Put
Смена	CHANGE REFERENCE DATA	Change
Первичная генерация	GENERATE KEY PAIR[6]	Generate Key
Перегенерация	GENERATE KEY PAIR[7]	Regenerate Key
Использование	PSO, VERIFY, AUTHENTICATE, SM	Use
Разблокирование	RESET RETRY COUNTER	Unblock
Прочитать открытый ключ	GENERATE KEY PAIR	Read Public Key
Прочитать свойства ключа	SELECT[8]	Read properties
Получить оставшееся число попыток предъявления	VERIFY	Read statistics

[5] Загрузка снаружи должна быть специально разрешена для конкретного ключа.

[6] В данной версии ОС ключ RSA не может быть сгенерирован на карте.

[7] В данной версии ОС ключ RSA не может быть повторно сгенерирован на карте.

[8] Следует проанализировать блок FCP, возвращаемый командой

Тип ключа (тип криптоалгоритма, с которым ассоциируется ключ):

Код (hex)	Алгоритм
00	Пароль
01	ГОСТ 28147-89
02	DES
03	3DES
21	ГОСТ Р 34.10 - 2001
22	RSA[9]

Алгоритмы хэширования:

Код (hex)	Алгоритм
11	хэш ГОСТ Р 34.11-94
12	хэш SHA-1

Флаги назначения ключа:

Битовая маска 87654321	Смысл
00000001	использование для SM
00000010	использование для внешней аутентификации
00000100	использование для внутренней аутентификации
00001000	использование для криптографических сервисов
00010000	требование диверсификации ключа
00100000	требование генерации сессионного ключа
01000000	RFU, всегда должен устанавливаться = 0
10000000	для симметричных алгоритмов всегда должен устанавливаться = 0 для асимметричных алгоритмов - разрешение загрузки секретного ключа командой CHANGE REFERENCE DATA

см. также свойства ключа в FCP (п.10.1).

5.3.2 Файлы правил разграничения доступа

Файлы ARF предназначены для хранения ***сложных*** правил разграничения доступа (см. п. 6.3).

Правила хранятся в формате Simple-TLV. Номер правила выступает в качестве тэга записи.

Примечание. *Правил с номерами 00 и FF, которые запрещены в ISO в качестве тэгов, не существует, т.к. эти значения имеют особый смысл – доступ всегда разрешен и всегда запрещен.*

В ARF не может быть двух правил с одинаковым номером. Модификация правил запрещена.

Основные операции над ARF:

Операция	Команда	Право доступа
Запись правила	PUT DATA	Put (свободно на фазе инициализации файла)
Чтение правила	GET DATA	Get

[9] RSA используется только в режиме формирования/проверки ЭЦП (ЭЦП формируется в соответствии с rfc2313)

5.3.3 Файлы хранения SE

SEF предназначены для хранения шаблонов настроек SE (CRT). С каждым SEF ассоциирован идентификатор SE. При активизации SE производится поиск всех ключей, на которые есть ссылки, причем поиск ключа идет от директории, где находится SEF и выше. Если хотя бы один ключ не найден, SE не может быть активизирован[10].

Основные операции над SE:

Операция	Команда	Право доступа
Запись/изменение SE	MSE-STORE	Store (свободно на фазе инициализации файла)
Активизация SE	MSE-RESTORE	Restore
Удаление SE	MSE-ERASE	Store
Динамическое изменение SE	MSE-SET	Free

6 СИСТЕМА РАЗГРАНИЧЕНИЯ ДОСТУПА

В данной версии "Руководства…" Раздел №6 приводится в сокращенном варианте. Полная версия руководства предоставляется по индивидуальному запросу mail@smart-park.ru

В основу системы разграничения доступа положен дискреционный принцип разграничения доступа, т.е. каждый файл данных имеет ***атрибуты*** доступа, определяющие ***условия*** осуществления различных ***видов*** доступа со стороны ***субъектов*** доступа.

Субъект аутентифицируется посредством доказательства знания определенного секрета. В результате такого доказательства в ОС устанавливается ***санкция***. Знание нескольких секретов позволяет субъекту доступа установить несколько разных санкций. Факт установки совокупности разных санкций можно оценить посредством вычисления логического выражения, состоящего из операторов ***И*** / ***ИЛИ***, в котором операндами являются санкции. Каждая установленная санкция в этом выражении представляет логическую величину TRUE, не установленная – FALSE.

Политика безопасности ОС предполагает, что запрещены все действия кроме явно разрешенных.

6.1 Санкции

Санкция – это разрешение на проведение некоторых действий.

Субъект доступа может выдавать санкции одним из следующих способов:

- Доказательством знания ключа или пароля в ходе процедуры аутентификации.
- Доказательством знания ключа, используя его для вычисления имитовставки передаваемых данных в режиме SM.
- Посредствам агента, представленного загруженным на карту исполняемым кодом расширения.

[10] Внимание! При формировании SE в памяти командами MSE-SET поиск ключа ведется от текущей в тот момент директории, которая может не совпадать с директорией, где расположен SEF.

Санкции, выданные в процессе аутентификации, действуют в течение всей сессии.

Санкции, выданные в результате применения SM, действуют только в течение выполнения команды.

Программные агенты (расширения) могут устанавливать как постоянные, так и одноразовые (на 1 команду) санкции.

Каждая санкция имеет 8-битовый идентификатор.

У санкций ассоциированных с ключами старший бит должен быть сброшен (т.е. значение в диапазоне 1..127).

Идентификатор ***0*** **зарезервирован**.

С каждым файлом ключа ассоциированы идентификаторы двух санкций, одна из которых автоматически выдается при предъявлении данного ключа, а вторая – при использовании SM. Эти ассоциированные с ключом санкции могут совпадать. Идентификаторы всех санкций должны быть уникальны – это автоматически контролируется средствами ОС. Санкции с нечетными идентификаторами могут использоваться в прямых правилах доступа (см. п. 6.3).

6.2 Атрибуты доступа

Как отмечалось ранее, атрибуты доступа определяют условия осуществления различных видов доступа. Атрибуты доступа зависят от типа файла. Для описания атрибутов доступа используется собственный формат ОС, в котором:

а) каждому виду доступа соответствует отдельный 1-байтовый атрибут;

б) каждый 1-байтовый атрибут содержит номер соответствующего правила доступа.

Номер правила доступа имеет следующие значения:

- 00 – доступ безусловно разрешен,
- четные числа в диапазоне 2..FEh означают сложные правила доступа, заданные в ARF,
- нечетные числа в диапазоне 1..7Fh означают прямое правило доступа. В этом случае номер правила непосредственно соответствует номеру санкции, ассоциированной с ключом,
- нечетные числа в диапазоне 81h..FDh – прямое правило доступа, где номер правила является номером санкции выдаваемой программным агентом (расширением) или кодом ОС,
- FFh – доступ безусловно запрещен.

При создании файла достаточно передать только те атрибуты доступа, которые используются для ключа создаваемого типа:

- для паролей и симметричных ключей это атрибуты доступа 0-6
- для асимметричных ключей без секретной части пары – атрибуты 0-7
- для асимметричных пар – атрибуты 0-9.

Атрибуты доступа DF (MF)[11]

Номер байта	Вид доступа	Номер байта	Вид доступа
	Activate File		Put Context
	Deactivate File		Create Child File
	Delete File		

[11] Для MF вид доступа Deactivate File и Delete File всегда принимает одинаковое значение.

Атрибуты доступа BF

Номер байта	Вид доступа	Номер байта	Вид доступа
	Activate File		Read
	Deactivate File		Update
	Delete File		Write (OR)

Атрибуты доступа FRF, VRF

Номер байта	Вид доступа	Номер байта	Вид доступа
	Activate File		Read
	Deactivate File		Update
	Delete File		Append

Атрибуты доступа CRF

Номер байта	Вид доступа	Номер байта	Вид доступа
	Activate File		Read
	Deactivate File		Update/Append
	Delete File		

Атрибуты доступа TF

Номер байта	Вид доступа	Номер байта	Вид доступа
	Activate File		Get
	Deactivate File		Put
	Delete File		

Атрибуты доступа KF

Номер байта	Вид доступа	Номер байта	Вид доступа
	Activate File		Put
	Deactivate File		Change
	Delete File		Unblock
	Use		*Read public key*
			Generate
			Regenerate

Атрибуты доступа ARF

Номер байта	Вид доступа	Номер байта	Вид доступа
	Activate File		Get
	Deactivate File		Put
	Delete File		

Атрибуты доступа SEF

Номер байта	Вид доступа	Номер байта	Вид доступа
	Activate File		Store /Erase
	Deactivate File		Restore
	Delete File		

6.3 Правила доступа

Правила доступа бывают прямые или сложные.

Прямые правила доступа содержат один идентификатор санкции и записываются непосредственно в атрибут доступа.

Прямые правила доступа всегда имеют **нечетные** номера.

Проверка прямого правила реализуется как контроль того, была ли предъявлена соответствующая санкция с помощью некоторого ключа.

Сложные правила доступа представляют собой логические выражения, в качестве аргументов которых применяются идентификаторы санкций. Если санкция с неким идентификатором установлена, считается, что аргумент равен TRUE, иначе – FALSE.

Само логическое выражение всегда построено в виде дизъюнкции конъюнкций аргументов.

$R = (Arg^1_1 \text{ AND } \dots \text{ AND } Arg^1_{N1}) \text{ OR } \dots \text{ OR } (Arg^m_1 \text{ AND } \dots \text{ AND } Arg^m_{Nm})$

Сложные правила доступа всегда имеют **четные** номера.

Пример:

Предположим, нам необходимо рассчитать сложное правило по следующему выражению:

$$(S_0 \text{ AND } S_3 \text{ AND } S_4) \text{ OR } (S_1 \text{ AND } S_4) \text{ OR } (S_7)$$

где S0 , S1 , S3, S4, S7 - это различные санкции (прямые от ключей и паролей, другие сложные – не имеет значения).

Тогда требуемое сложное правило запишется так:

```
(3 0 3 4)  (2 1 4)  (1 7)
```

Здесь каждая первая величина в скобках обозначает количество аргументов конъюнкции (AND).

Поиск ***сложных*** правил доступа осуществляется от директории, в которой находится файл, и выше до MF. Если правило не найдено, то считается, что соответствующий вид доступа запрещен.

В файловой системе может быть множество файлов ARF. Они могут располагаться как в одной, так и в разных директориях. Если правило не найдено в ближайшем ARF, то поиск продолжается выше, в родительском DF. Таким образом, более близкий к файлу ARF имеет приоритет. Если несколько ARF находятся в одной директории, то порядок их обхода не определен.

ОС запрещает добавлять в ARF правило с номером, который уже присутствует в одном из ARF, доступных из данной директории (т.е. находящихся в том же DF или выше). Это позволяет предотвратить маскирование правил уровня родительского DF. Вместе с тем, в ARF можно добавить правило с номером, который уже существует в одном из ARF во

вложенных или не пересекающихся директориях[12].

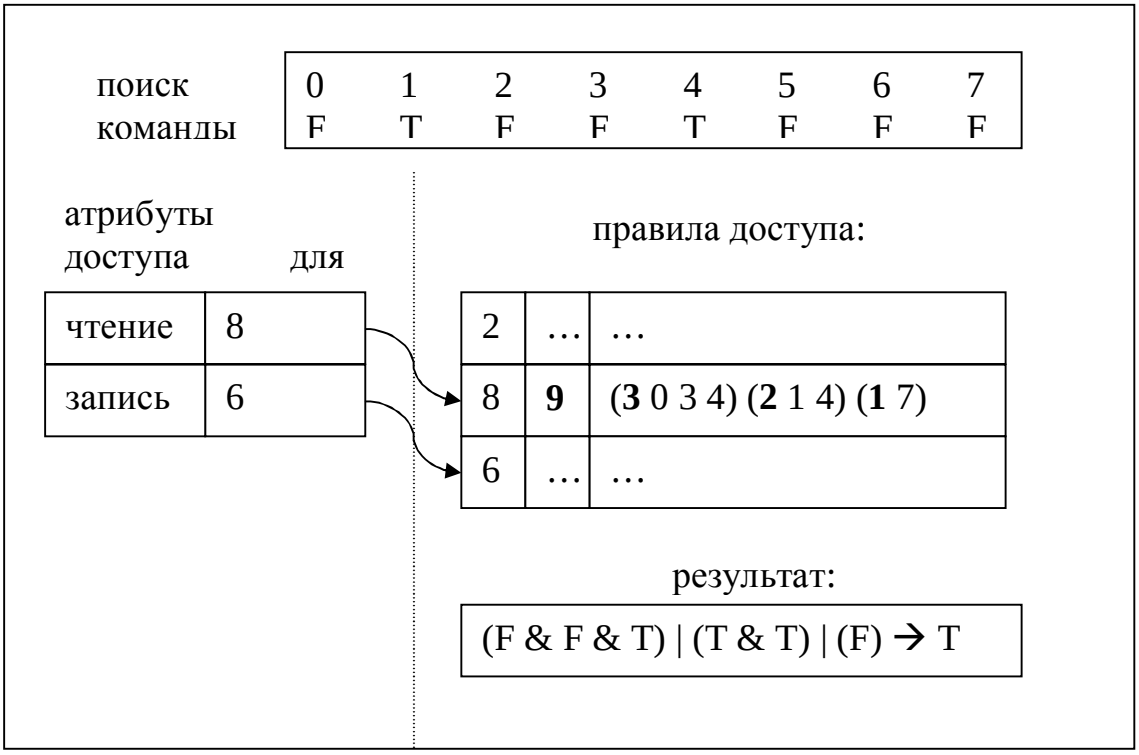
При разработке приложения карты важно помнить, что прямые правила всегда имеют четные номера. Если, по какой-либо причине, необходимо использовать санкцию ключа с четным значением, то необходимо создать соответствующе сложное правило, пусть даже состоящее всего из одного аргумента: $R = Arg_{even}$.

6.4 Авторизация доступа

При всяком обращении к объекту файловой системы выполняется проверка прав доступа:

- Берется атрибут доступа объекта, соответствующий виду доступа, и из него извлекается ссылка на правило доступа.
- Если правило *сложное*, то производится поиск правила доступа вверх по иерархии объектов.
- Вычисляется правило на основе установленных санкций и проверяется его результат.
- Если правило *простое*, то проверяется – установлена ли санкция на основе соответствующего ключа.

Пример авторизации для операции чтения (сложное правило)



[12] это не приводит к маскированию в том смысле, что на момент создания ARF во вложенном DF, этот ARF еще ничего не маскировал.

7 КРИПТОГРАФИЧЕСКИЕ ОПЕРАЦИИ И ПРОТОКОЛЫ

В данной версии "Руководства…" Раздел №7 приводится в сокращенном варианте. Полная версия руководства предоставляется по индивидуальному запросу mail@smart-park.ru

Карта использует криптографические операции для обеспечения:

- Аутентификации
- Защищенного обмена сообщениями
- Криптографических сервисов

Криптографические ключи размещаются в файлах ключей KF, где также хранятся атрибуты ключа.

Управление криптографическими операциями осуществляется при помощи Secure Environment.

..

7.1 Secure Environment

Secure Environment (SE) – набор настроек криптографических операций.

На карте может присутствовать множество различных SE, хранящихся в специальных файлах SEF. С каждым SEF связан 1-байтовый идентификатор.

Управление SE осуществляется командой MSE (см. п.9.7.1). Команда MSE имеет 4 режима работы:

…

…

Полная версия руководства предоставляется по индивидуальному запросу mail@smart-park.ru

7.2 Аутентификация

Методы аутентификации

Режим аутентификации	id ключа в команде	Ген-ция сессион. ключа	Разреш. внешней аут-ции	Разреш. внутренней аут-ции	Шаблоны настроек SE
Аутентификация по паролю	*	-	+	-	AT/VERIFY
Внешняя аутентификация	*	-	+	-	AT/VERIFY (+HT)
Внутренняя аутентификация	*	-	-	+	AT/CALC (+HT)
Взаимная аутентификация по симметричной схеме	-	*	+	-	KAT/CALC|VERIFY
Взаимная аутентификация по несимметричной схеме	-	*	+	-	KAT/CALC|VERIFY

Легенда:

+ обязательный элемент

* опциональный элемент

- применение невозможно

Все методы взаимной аутентификации допускают генерацию сессионного ключа, если это требуется в атрибутах ключа.

Во всех режимах аутентификации можно использовать настройки текущего SE, однако некоторые режимы аутентификации допускают непосредственную ссылку на ключ без предварительной установки шаблона настроек АТ. В таких командах шаблон АТ устанавливается автоматически и сохраняется после завершения команды. . Если при этом корректно указан существующий ключ, то шаблон устанавливается вне зависимости от результата аутентификации.

В атрибутах ключа, используемого для аутентификации, должны быть разрешены соответствующие виды аутентификации: внутренняя и/или внешняя. Сессионный ключ наследует атрибуты мастер-ключа в случае симметричного алгоритма и атрибуты секретного ключа в случае асимметричного алгоритма.

Далее описываются протоколы аутентификации для различных режимов:

7.2.1 Аутентификация по паролю

Аутентификация производится на основе сравнения переданных данных с паролем, хранящимся в файле. Пароль должен иметь размер **8** байт[13].

Примечание. *Преобразование фактического значения пароля к блоку данных требуемого размера осуществляется внешним приложением. Можно использовать вычисление хэша или какой-либо способ выравнивания данных.*

Терминал	Команда	Карта
ввод пароля P		
→	VERIFY (P)	
		проверка пароля P = P’

7.2.2 Внешняя аутентификация

Карта устанавливает – владеет ли терминал требуемым секретом. Терминал получает от карты сгенерированное ею случайное число, шифрует его или вычисляет ЭЦП. Карта проверяет корректность результата.

При использовании симметричного алгоритма, на карте должен существовать секретный ключ. При использовании схемы с открытым ключом, открытый ключ должен быть заранее загружен на карту.

При использовании симметричного алгоритма:

[13] Ограничение будет устранено в будущих версиях ОС

Терминал	Команда	Карта
	GET CHALLENGE ()	
		генерирует СЧ R
	Response(R)	
C = Enc(R)		
	EXTERNAL AUTHENTICATE (C)	
		C ?= Enc(R)

При использовании асимметричного алгоритма:

Терминал	Команда	Карта
	GET CHALLENGE ()	
		генерирует СЧ R
	Response(R)	
C = Enc(H(R))		
	EXTERNAL AUTHENTICATE (C)	
		H(R) ?= Dec(C)

Enc – шифрование или вычисление ЭЦП
Dec – расшифрование или проверка ЭЦП
H – вычисление хэша

7.2.3 Внутренняя аутентификация

Терминал устанавливает – владеет ли карта требуемым секретом. Терминал генерирует случайное число, которое карта шифрует или вычисляет на нем ЭЦП. Терминал проверяет корректность результата.

При использовании симметричного алгоритма терминал должна знать секретный ключ. При использовании схемы с открытым ключом, открытый ключ карты должен быть предварительно прочитан с карты и загружен на терминал[14].

При использовании симметричного алгоритма:

Терминал	Команда	Карта
генерирует СЧ R		
	INTERNAL AUTHENTICATE (R)	
		C=Enc(R)
	Response(C)	
C ?= Enc(R)		

[14] С выполнением необходимых проверок.

При использовании симметричного алгоритма:

Терминал	Команда	Карта
генерирует СЧ R		
	INTERNAL AUTHENTICATE (R)	
		C=Enc(H(R))
	Response(C)	
H(R) ?= Dec(C)		

Enc – шифрование или вычисление ЭЦП
Dec – расшифрование или проверка ЭЦП
H – вычисление хэша

7.2.4 Взаимная аутентификация по симметричной схеме

Два субъекта проверяют друг друга – владеют ли они требуемым секретом. В качестве субъектов могут выступать две аналогичные карты. Стороны обмениваются случайными числами, на основе них генерируют сессионный ключ и проверочные криптограммы и обмениваются проверочными криптограммами.

При необходимости, в контексте устанавливается сессионный ключ.

…

…

Полная версия руководства предоставляется по индивидуальному запросу mail@smart-park.ru

7.2.5 Взаимная аутентификация по несимметричной схеме

Стороны обмениваются случайными числами, на основе них генерируют сессионный ключ и проверочные криптограммы. Карта проверяет криптограмму ответной стороны, после чего передает ей собственную криптограмму.

При необходимости, в контексте устанавливается сессионный ключ.

Аутентификация по несимметричной схеме не допускает использование в качестве ответной части аналогичной карты.

…

…

Полная версия руководства предоставляется по индивидуальному запросу mail@smart-park.ru

7.3 Защищенный обмен сообщениями

Защищенный обмен сообщениями (SM – Secure Messaging) обеспечивает криптографическую аутентификацию и/или сокрытие передаваемых данных. Параметры криптографических операций определяются текущим SE.

Для передаваемых и принимаемых данных используются одинаковые параметры.

Заголовок команды всегда участвует в формировании имитовставки.

…

…

Полная версия руководства предоставляется по индивидуальному запросу
mail@smart-park.ru

7.4 Криптографические сервисы

Криптографические сервисы реализуются командой PSE.

Для работы криптографических сервисов используется текущий SE.

В атрибутах ключей должно быть разрешено применение ключа для криптографических сервисов.

Следующая таблица показывает, какие шаблоны настроек SE используются и какие элементы в них присутствуют

Настройки SE для криптографических сервисов (PSO)

Сервис	Шаблон / назначение шаблона	Ссылка на ключ (83)	Ссылка на секр. ключ(84)	Синхро-посылка (87)
шифрование				
расшифрование				
вычисление имитовставки				
проверка имитовставки				
вычисление хэша				
вычисление ЭЦП				
проверка ЭЦП				
проверка сертификата				

Синтаксис команды см. в разделе «Криптографические команды» (п.9.7).

8 ПРИЛОЖЕНИЯ

Приложения для Изделия разрабатываются, инициализируются, персонализируются и эксплуатируются в рамках фазы жизни "Эксплуатация". Функциональные права доступа на создание приложения (права доступа к MF) определяются поставщиком карт при их переводе на фазу эксплуатации.

Загрузка приложений должна выполняться на выделенных для этого рабочих местах, к которым ограничен доступ посторонних лиц и приняты меры по защите от НСД через каналы связи, входящие в состав рабочих мест.

9 КОМАНДЫ ОС

9.1 Команды управления файлами

Команды управления файлами предназначены для создания, удаления файлов и управления жизненным циклом файлов.

Все команды данной группы кроме CREATE FILE позволяют выбрать файл, к которому будут применяться. Определено несколько режимов выбора файла, которые перечислены ниже:

P1	режим выбора файла (1) 00 – применить команду к текущему файлу (Lc=0) (2) 00 – выбор файла по идентификатору (Lc=2) (3) 01 – выбор вложенного DF по идентификатору (4) 02 – выбор вложенного EF по идентификатору (5) 08 – выбор файла по абсолютному пути (от MF) (6) 09 – выбор файла по относительному пути (от текущего DF)		
P2	00		
Lc, данные	режим	Lc	данные
	1	0	—
	2,3,4	2	идентификатор файла
	5,6	n (четно)	путь (конкатенация id)

Таблица 1

Отличия от выбора файла в команде SELECT заключаются в следующем:

- Иначе обозначается применение команды к текущему файлу (P1:P2= 00:00).
- Отсутствует режим последовательного перебора файлов.
- Отсутствует выбор родительского DF.
- Отсутствует выбор приложения по AID.

В остальном действуют те же правила, что и в команде SELECT.

Если выбираемого файла не существует, то текущим остается ранее выбранный файл.

9.1.1 CREATE FILE

Создание файла внутри текущего DF.
После успешного создания файла, он становится текущим файлом.
Если команда выполнена неудачно, текущий файл не изменяется.

тип команды	in (case 3)

типы файлов:

DF (MF)	BF	FRF	VRF	CRF	TF	KF	ARF
+	*	*	*	*	*	*	*

формат команды:

CLA	
INS	
P1	
P2	
Lc	
данные	
Le	—
ответ	—

характерные статусы ошибок:

6984	Данные команды имеют недопустимые значения
6A80	Данные команды имеют недопустимый формат
6A84	Недостаточно свободного места на карте или в одном из объемлющих DF
6A88	Необходимые данные отсутствуют
6A89	Файл с заданным идентификатором уже существует

Идентификатор создаваемого файла должен быть уникален в пределах текущего DF (в том числе не должен совпадать с идентификатором самого текущего DF).

9.1.2 ACTIVATE FILE

Перевести файл на фазу жизни «***использование***».

Перед выполнением команды файл должен находиться на фазе жизни «***инициализация***» или «***блокирование***».

тип команды	in (case 3)

типы файлов:

DF (MF)	BF	FRF	VRF	CRF	TF	KF	ARF
+	+	+	+	+	+	+	+

формат команды:

CLA	
INS	
P1	
P2	
Lc	
данные	
Le	—
ответ	—

характерные статусы ошибок:

6700	Неправильная длина входных данных
6984	Входные данные команды имеют недопустимые значения
6989	Операция неприменима к данной фазе жизни файла
6A82	Файл не найден
6A86	Неправильные значения параметров P1/P2

Если P1:P2=0000 и входные данные отсутствуют, то команда применяется к текущему файлу.

9.1.3 DEACTIVATE

Перевести файл на фазу жизни «***блокирование***».

Перед выполнением команды файл должен находиться на фазе жизни «***инициализация***» или «***использование***».

тип команды	in (case 3)

типы файлов:

DF (MF)	BF	FRF	VRF	CRF	TF	KF	ARF
+	+	+	+	+	+	+	+

формат команды:

CLA	
INS	
P1	
P2	
Lc	
данные	
Le	-
ответ	-

характерные статусы ошибок:

6700	Неправильная длина входных данных
6984	Входные данные команды имеют недопустимые значения
6989	Операция неприменима к данной фазе жизни файла
6A82	Файл не найден
6A86	Неправильные значения параметров P1/P2

Если P1:P2=0000 и входные данные отсутствуют, то команда применяется к текущему файлу.

9.1.4 DELETE FILE

Удалить файл.
Перед удалением файл может находиться на любой фазе жизни.
При удалении DF автоматически удаляются все вложенные файлы этого DF.

тип команды	in (case 3)

типы файлов:

DF (MF)	BF	FRF	VRF	CRF	TF	KF	ARF
+ (-)	+	+	+	+	+	+	+

формат команды:

CLA	
INS	
P1	
P2	
Lc	
данные	
Le	—
ответ	—

характерные статусы ошибок:

6700	Неправильная длина входных данных
6984	Входные данные команды имеют недопустимые значения
6986	Операция неприменима к текущему файлу
6A82	Файл не найден
6A86	Неправильные значения параметров P1/P2

Если P1:P2=0000 и входные данные отсутствуют, то команда применяется к текущему файлу.

Если во время удаления DF работа карты будет прервана, удаление вложенных файлов будет автоматически продолжено после рестарта карты.

После удаления файла текущим становится его родительский DF/MF.

MF удалить невозможно.

9.1.5 TERMINATE CARD USAGE

Полностью и безвозвратно заблокировать карту. Карта не возвращает статус, более не реагирует на сброс и ведет себя как при физическом выведении из строя.

Команда работает на всех фазах жизни карты. На фазе инициализации требуется предъявление административного ключа, а на фазе эксплуатации требуются права доступа на применение операции DEACTIVATE для MF[15].

Текущий файл может быть любым.

тип команды	case 1

формат команды:

CLA	
INS	
P1	
P2	
Lc	—
данные	—
Le	—
ответ	—

[15] Заметим, что для MF можно также применять команду DEACTIVATE – временное блокирование.

9.2 Команды навигации

9.2.1 SELECT

Выбор текущего файла.

тип команды	in/out (case 4)

формат команды:

CLA	00
INS	A4
P1	режим выбора файла (1) 00 – выбор файла по идентификатору (Lc=2) или (2) навигация по директории (см. P2) (Lc=0) (3) 01 – выбор DF под текущим DF по идентификатору (4) 02 – выбор EF под текущим DF по идентификатору (5) 03 – выбор родительского файла текущего DF (поле данных пусто) (6) 04 – выбор директории по имени (7) 08 – выбор файла по абсолютному пути (от MF) (8) 09 – выбор файла по относительному пути (от текущего DF)
P2	Биты 1,2 для режимов (2),(6): 00 – выбрать первый файл 10 – выбрать следующий файл Биты 3,4: 00 – вернуть FCI 01 – вернуть FCP 10 – вернуть FMD 11 – ничего не возвращать
Lc, данные	режим / Lc / данные 1,3,4 / 2 / идентификатор файла 2,5 / 0 6 / 0<=n<=16 / имя приложения (AID) 7,8 / n>0 (четно) / путь (конкатенация id)
Le	обычно 0 – все имеющиеся данные
ответ	FCI или FCP, или FMD, или ничего

характерные статусы ошибок:

6283	Выбранный файл блокирован
6700	Неправильная длина входных данных
6984	Входные данные команды имеют недопустимые значения
6A82	Файл не найден
6A86	Неправильные значения параметров P1/P2

По сравнению с ISO 7816-4 внесены следующие изменения:

- Последовательный перебор приложений с одинаковым именем делается не подачей одной и той же команды, а установкой P2[1..2] = 02
- Не поддерживаются режимы выбора последнего и предыдущего файла P2[1..2]=01,03

Если выбираемого файла не существуют, то текущим остается ранее выбранный файл.

Зарезервированные идентификаторы для P1=00:

- идентификатор `3F00` зарезервирован за MF;
- идентификатор `3FFF` обозначает текущий DF;
- идентификатор `0000` обозначает текущий файл (EF или DF).

Если текущий файл – DF, то действуют следующие правила:

- Анализируются зарезервированные идентификаторы, причем 3FFF означает сам текущий DF.
- При выборе первого файла (только для SELECT) (P1=00 P2=00), текущим становится первый файл вложенной директории, т.е. происходит «погружение» в DF.
- При выборе следующего файла (P1=00, P2=03), текущим становится следующий за DF файл, расположенный на том же уровне, т.е. «погружения» не происходит. Таким образом, всю директорию можно просмотреть, последовательно выбирая следующий файл, при этом не будет производиться погружение во вложенные директории.
- При выборе файла по идентификатору поиск производится в следующем порядке:
  - внутри DF
  - на одном уровне с DF
  - анализируется идентификатор родительского DF
- При выборе родительского DF (P1=03) выбирается родительский файл данного DF.

Если текущий файл – EF, то действуют следующие правила:

- Анализируются зарезервированные идентификаторы.
- При выборе файла по идентификатору поиск производится в следующем порядке:
  - внутри текущего DF
  - анализируется идентификатор текущего DF
- При выборе родительского DF (P1=03) выбирается родительский файл текущего DF (т.е. осуществляется выбор «дедушки»)

Выбор по пути равносилен последовательному выполнению нескольких команд SELECT в режиме выбора по идентификатору (P1=0). При выборе файла с указанием пути, начало пути (MF или текущий DF не указываются). Выбор по абсолютному пути выглядит так, как если бы в начало пути был добавлен идентификатор 3F00, а по относительному пути – идентификатор 3FFF.

Имя DF (AID) находится в контексте DF. При выборе DF по имени, поиск начинается с MF и производится по всем директориям карты с погружением в поддиректории. В режиме P2=2, поиск продолжается, начиная со следующего DF в порядке обхода. Поиск производится по частичному совпадению, т.е. достаточно, чтобы совпали первые Lc байт имени. Если Lc = 0, то будет выбран первый встречный DF, вообще имеющий AID.

Содержание FCI, FCP и FMD соответствует ISO 7816-4 (см. раздел «Структуры данных на карте»).

FCI является конкатенацией элементов данных, входящих в FCP и FMD[16]. Фактически, FMD для DF – содержание контекста DF; для EF FMD не определен.

При запросе вывода FMD или FCI будет возвращено столько тэгов контекста DF, чтобы суммарная длина всех выдаваемых данных не превысила 255 байт.

В случае указания отличного от нуля значения уровня выдачи информации по команде SELECT (объект с тэгом 8D шаблона A5 в FCP) для родительского файла выбранного файла объем выводимой информации будет соответствующим образом редуцирован.

[16] В предыдущих версиях ОС FCI рассматривался как конкатенация шаблонов FCP и FMD, а не только их данных.

9.3 Команды работы с данными BER-TLV

Команды данной категории предназначены для работы с файлами TLV-записей (TF) и контекстами директорий.

См. также раздел «Команды работы с правилами доступа» п. 9.6.

типы файлов (кроме случая специальных тэгов – см. описание команд):

DF (MF)	BF	FRF	VRF	CRF	TF	KF	ARF
+	—	-	-	-	+	-	-

Команды имеют по 2 разновидности: с четным INS и нечетным INS.

В случае четного INS, тэг записи кодируется в P1:P2. В соответствии с форматом BER-TLV, P1:P2 принимают значения в диапазонах:

- 00:40..00:FE – для однобайтовых тэгов.
- 40:00..FFFF – для 2-х байтовых тэгов.

Если выбранная запись является шаблоном, она считывается полностью. Установка шаблона в качестве текущего контекста для дальнейшего считывания из него примитивных элементов данных не поддерживается из-за двусмысленности реализации.

В случае нечетного INS, P1:P2 – идентифицируют файл:

- 00:00 – текущий файл
- если старшие 11 бит =0, а младшие 5 бит образуют значение от 1 до 30, то младшие 5 бит – короткий идентификатор файла.
- в остальных случаях, P1:P2 – полный идентификатор файла.

При этом ссылка на конкретную запись содержится в данных – см. описание команд.

9.3.1 GET DATA

Прочитать TLV-запись из файла TF или контекста директории.

Примечание. *Если запись слишком велика для передачи в одной команде, она может быть прочитана в несколько приемов командами GET RESPONSE – не реализовано в v1.0*

При использовании четного INS (INS=CA) несколько тэгов зарезервированы для получения специальной информации:

тэг	значение
5F51	Answer-to-Reset
5F52	байты истории (historical bytes) ATR
00FF	полностью считать контекст текущей директории
0000	полностью считать содержимое текущего TF

При использовании тэга 0000 текущим должен быть файл типа TF, а для остальных специальных тэгов – любой файл. При чтении зарезервированных тэгов текущий файл не меняется.

Байты истории ATR могут быть модифицированы командой PUT DATA на значение пользователя. В этом случае пользовательское значение хранится в контексте MF и считывается оттуда.

тип команды	in/out (case 4)

формат команды:

CLA	00
INS	CA или CB
P1:P2	
Lc	размер данных
данные	если INS=CA – ничего если INS=CB – список запрашиваемых записей (см. ниже)
Le	0 или N
ответ	Le=0 – запрошенные записи целиком или первые 256 байт от них Le=N – первые N байт запрошенных записей

характерные статусы ошибок:

6700	Неправильная длина входных данных
6984	Данные команды имеют недопустимые значения
6A80	Данные команды имеют недопустимый формат
6A82	Файл не найден
6A86	Неправильные значения параметров P1/P2
6A88	Необходимые данные отсутствуют

Если INS четно, то поле данных запроса пусто, а назад возвращается содержание (поле V) запрошенной записи.

Если INS нечетно, в запросе передается список заголовков запрашиваемых записей, а в ответе – список (конкатенация) запрошенных записей в формате BER-TLV.

Список заголовков оформляется в виде BER-TLV записи и представляет собой одну из

следующих конструкций, различаемых по тэгу:

конструкция	тэг	устройство запроса	устройство ответа
список тэгов	5C	Конкатенация тэгов запрашиваемых записей (без длин)	список запрошенных записей в формате BER-TLV
список заголовков	5D	Конкатенация пар TL запрашиваемых записей	список усеченных записей в формате BER-TLV

Если в списке заголовков (тэг 5D) для какого либо входящего в него тэга указывается длина L=0, то соответствующая запись не усекается. Если длина L>0, то передается не больше L байт соответствующей записи.

Пусть в TF лежат следующие записи:

Tag	Length	Value
5F 21	01	11
7F 22	07	45 01 01 46 02 02 02
41	02	33 33

Запрос (список тэгов):

```
5C 05 5F21 7F22 41
```

Ответ:

```
5F21 01 11  7F22 07 45010146020202  41 02 3333
```

Запрос (список заголовков):

```
5D 08  5F21 05  7F22 00  41 01
```

Ответ:

```
5F21 01 11  7F22 07 45010146020202  41 01 33
```

9.3.2 PUT DATA

Занести TLV-запись в файл TF или контекст директории.

Если запись с таким тэгом уже присутствует, то она замещается новой записью при условии равенства длин. Если длина новой записи отличается, то возникает ошибка. Таким образом, в файле может быть только одна запись с определенным тэгом.

Чтобы заместить уже существующую запись, требуется право доступа на вид доступа UPDATE, а чтобы добавить новую запись – APPEND.

При использовании четного INS (INS=DA) тэг 5F52 зарезервирован для записи пользовательского значения байтов истории (historical bytes) ATR. В этом случае пользовательское значение записывается в контекст MF (длина байтов истории не должна превышать 15 байт). Текущий файл при выполнении этой команды с зарезервированным значением тэга может быть любым и не меняется после выполнения команды.

Чтобы иметь возможность менять байты истории на пользовательское значение, контекст MF должен создаваться достаточного размера для хранения самих байтов истории и Ber-TLV заголовка (3 байта).

Будучи единожды записаны, байты истории могут быть изменены на другое значение, но без изменения длины.

Запись байтов истории требует права на доступ по записи к контексту MF.

тип команды	in (case 3)

формат команды:

CLA	00
INS	DA или DB
P1:P2	
Lc	размер данных
данные	если INS=DA, содержание записи (поле V) если INS=DB, список (конкатенация) записей в формате BER-TLV
Le	—
ответ	—

характерные статусы ошибок:

6700	Неправильная длина входных данных
6984	Данные команды имеют недопустимые значения
6A80	Данные команды имеют недопустимый формат
6A82	Файл не найден
6A86	Неправильные значения параметров P1/P2
6A88	Необходимые данные отсутствуют

Если в P1:P2 при INS=DA содержится тэг соответствующий шаблону, либо при INS=DB среди списка записей содержатся записи-шаблоны, то таковые помещаются в файл как единое целое без дальнейшего разбора на входящие в шаблон записи.

9.4 Команды работы с бинарными файлами

В командах работы бинарными файлами (BF) поддерживаются следующие режимы адресации к данным:

P1	bit 7 = 1 bit 0 .. bit 4 – короткий идентификатор файла bit 7 = 0 bit 0 .. bit 6 - MSB адреса
P2	LSB адреса

Таблица 2

Нечетные INS не поддерживаются.

BF заполняется нулевыми байтами (со значением 00h) при создании.

типы файлов:

DF (MF)	BF	FRF	VRF	CRF	TF	KF	ARF
-	+	-	-	-	-	-	-

9.4.1 READ BINARY

Команда читает данные из BF по указанному смещению. Если команда содержит короткий идентификатор файла, то файл предварительно выбирается и остается текущим после завершения команды.

тип команды	out (case 2)

формат команды:

CLA	00
INS	B0
P1:P2	см. Таблица 2
Lc	—
данные	—
Le	кол-во байт для чтения
ответ	данные

характерные статусы ошибок:

6A82	файл не найден
6B00	неверные параметры P1-P2 – смещение за границей файла

В случае указания Le=00 производится чтение до конца файла, либо до достижения максимального размера коммуникационного буфера (256/246 байт без использования SM по контактному/бесконтактному интерфейсам и меньший размер при использовании SM).

9.4.2 UPDATE BINARY

Команда записывает данные в BF по указанному смещению. Если команда содержит короткий идентификатор файла, то файл предварительно выбирается и остается текущим после завершения команды.

тип команды	in (case 3)

формат команды:

CLA	00
INS	D6
P1:P2	см. Таблица 2
Lc	количество байт для записи
данные	данные
Le	—
ответ	—

характерные статусы ошибок:

6700	неверная длина входных данных – ноль байт для записи
6A82	файл не найден
6B00	неверные параметры P1-P2 – смещение за границей файла

9.4.3 WRITE BINARY

Команда производит побитовую операцию OR над передаваемыми данными и предыдущим содержимым файла. Записывается Lc байт по указанному смещению. Если команда содержит короткий идентификатор файла, то он предварительно выбирается и остается текущим после завершения команды.

тип команды	in (case 3)

формат команды:

CLA	00
INS	D0
P1:P2	см. Таблица 2
Lc	количество байт для записи
данные	данные
Le	—
ответ	—

характерные статусы ошибок:

6700	неверная длина входных данных – ноль байт для записи
6A82	файл не найден
6B00	неверные параметры P1-P2 – смещение за границей файла

9.5 Команды работы с файлами записей

Команды данной группы применимы к файлам записей фиксированной (FRF) и переменной (VRF) длины, а также к циклическим файлам записей фиксированной длины (CRF).

типы файлов:

DF (MF)	BF	FRF	VRF	CRF	TF	KF	ARF
-	—	+	+	+	-	-	-

Если команда содержит короткий идентификатор файла, то файл предварительно выбирается и остается текущим после завершения команды.

К записям можно осуществлять доступ по номеру, по тэгу (для VRF) и по указателю на текущую запись. При доступе по номеру указатель на текущую запись не сдвигается.

Сразу после явного выбора файла (в том числе с использованием обращения по короткому идентификатору) указатель на текущую запись выставляется на первую запись независимо от того, был ли ранее этот файл текущим, причем в этом случае доступ к следующей записи равносилен доступу к первой записи.

В файлах с линейной организацией первой считается самая ранняя запись, а в циклических файлах – самая поздняя запись.

Записи нумеруются от 1. Номер 0 обозначает текущую запись.

В файлах VRF записи хранятся в формате SimpleTLV с полями Tag и Length размером 1 байт. В файле может быть несколько записей с одинаковыми тэгами. Тэги 0 и 255 зарезервированы. Поле данных команд добавления, обновления и чтения записей для файлов VRF всегда представляется формате SimpleTLV, то есть, первый байт – тэг, второй – длина последующих данных, далее – сами данные указанной длины.

Количество записей в файле любого типа ограничено числом **254**.

Размер каждой записи в файлах фиксированной длины не больше **255** байт, а в файлах записей переменной длины – не больше **254** байт (т.к. значение длины 255 зарезервировано для 3-х байтовых длин).

Если файл пуст, то попытка прочитать запись любым способом повлечет ошибку (как в линейном, так и в циклическом файле). Попытка чтения записей «по второму кругу» в циклическом файле также приводит к ошибке.

Удаление записей не предусмотрено. Обновление (переписывание) записей возможно только в случае совпадения длины старой и новой записи.

Заголовок команд в общем случае кодируется следующим образом (кроме команды APPEND RECORD в случае FRF, CRF, VRF):

P1	00 - текущая запись 1..254 - номер записи, если младшие 3 бита P2 имеют значение 4 - тэг записи для VRF, если младшие 3 бита P2 имеют значение 0,2 FF – RFU
P2	старшие 5 бит: 0 - текущий файл 1..30 – короткий идентификатор файла 31 - RFU младшие 3 бита: 0 – первая запись [для VRF – с тэгом P1, если P1≠0] 2 – следующая запись [для VRF – с тэгом P1, если P1≠0] 4 – ссылка на запись по номеру из P1 1,3,5..7 – не поддерживается

Таблица 3

Ниже приводятся примеры кодирования заголовков команд для разных типов файлов.
Для линейного файла записей постоянной длины (FRF):
00 00 - прочитать первую запись
00 02 - прочитать следующую запись
00 04 - прочитать текущую запись
xx 04 - прочитать запись #xx

Для циклического файла записей (CRF):
00 00 - прочитать последнюю (самую позднюю) запись
00 02 - прочитать предыдущую запись
00 04 - прочитать текущую запись
xx 04 - прочитать запись #xx от конца

Для файла записей переменной длины (VRF):
00 00 - прочитать первую запись
00 02 - прочитать следующую запись
00 04 - прочитать текущую запись
xx 00 - прочитать первую запись с тэгом xx
xx 02 - прочитать следующую запись с тэгом xx
xx 04 - прочитать запись #xx

Нечетные INS, обеспечивающие доступ к записи по частям, не поддерживаются.

9.5.1 APPEND RECORD

Добавление записи.
Для CRF – переписывание самой старой записи, если весь файл заполнен.

тип команды	out (case 2)

формат команды:

CLA	00
INS	E2
P1	00 – для FRF, CRF, VRF
P2	старшие 5 бит - см. Таблица 3 младшие три бита: 0 – для FRF, CRF, VRF
Lc	размер записи
данные	запись
Le	—
ответ	—

9.5.2 READ RECORD

Чтение записи.

тип команды	out (case 2)

формат команды:

CLA	00
INS	B2
P1:P2	см. Таблица 3
Lc	—
данные	—
Le	0 or N>0
ответ	Le=0 – запись целиком Le=N – первые N байт записи

9.5.3 UPDATE RECORD

Обновление записи.

тип команды	out (case 2)

формат команды:

CLA	00
INS	DC
P1:P2	см. Таблица 3
Lc	размер записи
данные	запись
Le	—
ответ	—

9.6 Команды работы с правилами доступа

Для работы с правилами доступа служат команды PUT DATA и GET DATA (см. раздел «Команды работы с данными BER-TLV» п. 9.3) правила доступа представлены в формате Simple-TLV и кодируются в соответствии с рекомендациями ISO 7816-4 для этих команд.

типы файлов:

DF (MF)	BF	FRF	VRF	CRF	TF	KF	ARF
-	—	-	-	-	-	-	+

9.6.1 GET DATA

Прочитать правило с заданным идентификатором.
Файл ARF должен быть текущим.

тип команды	out (case 2)

формат команды:

CLA	00
INS	CA
P1	02
P2	номер правила 02..FE
Lc	—
данные	—
Le	размер правила
ответ	содержание правила (без тэга и длины)

характерные статусы ошибок:

6A86	Неправильные значения параметров P1/P2
6A88	Необходимые данные отсутствуют

9.6.2 PUT DATA

Записать правило с заданным идентификатором.

Если правило с таким идентификатором уже присутствует в данном файле или в других ARF, в данной директории или выше, то будет ошибка.

Файл ARF должен быть текущим.

тип команды	in (case 3)

формат команды:

CLA	00
INS	DA
P1	02
P2	номер правила 02..FE
Lc	размер правила
данные	содержание правила (без тэга и длины)
Le	—
ответ	—

характерные статусы ошибок:

6A80	Данные команды имеют недопустимый формат
6A86	Неправильные значения параметров P1/P2
6A89	Правило с указанным идентификатором уже есть

9.7 Криптографические команды

Команды данной группы включают команду MANAGE SECURE ENVIRONMENT (MSE), описанную в стандарте ISO 7816-4 и все команды ISO 7816-8.

9.7.1 MANAGE SECURITY ENVIRONMENT

Команда MSE предназначена для управления криптографическим контекстом (SE), определяющим текущий режим работы SM (защищенного обмена сообщениями), криптографическим команд PSO и команд аутентификации.

Команда позволяет:

- Записывать SE в файл SEF (режим STORE)
- Устанавливать SE с указанным идентификатором в качестве текущего (режим RESTORE)
- Удалять записанный SE (режим ERASE)
- Устанавливать временные параметры текущего SE (режим SET)

Режим ***ERASE*** очищает SEF, но не влияет на состояние текущего SE в памяти.

Добавление шаблонов (кроме шаблона аутентификации AT) в режиме SET допустимо, только если соответствующие шаблоны с таким же назначением (см. P1) еще не установлены в текущем SE. Шаблон аутентификации AT разрешается менять неоднократно без необходимости сброса SE.

Чтобы сбросить SE необходимо подать команду ***RESTORE*** 0 (для загрузки «пустого» SE №0).

Для всех команд ***SET*** действуют права доступа на запись для того SE, который был активизирован последний раз.

тип команды	in (case 3)

формат команды:

CLA	00
INS	22
P1	• …
P2	
Lc	размер данных
данные	В режимах STORE, RESTORE, ERASE – ничего В режиме SET – CRT
Le	нет
ответ	ничего

В режиме SET команда MSE позволяет устанавливать временные параметры текущего SE:

тэг	CRT	значения
87	любой (не анализируется)	синхропосылка
94	любой кроме HT и DST	данные для диверсификации ключа

Для диверсификации ключа необходимо подать команду MSE в режиме SET, указав шаблон CRT, в котором фигурирует данный ключ и передав в тэге 94 данные для диверсификации (например, серийный номер).

Для указания синхропосылки для последующего использования командой PSO необходимо подать команду MSE в режиме SET, передав в тэге 87 значение синхропосылки. Тип шаблона CRT игнорируется и может быть пропущен, назначение шаблона – любое.

9.7.2 GENERATE KEY PAIR

Команда генерации ключевой пары.

тип команды	in/out (case 4)

формат команды:

CLA	00
INS	46
P1	0 – сгенерировать новую ключевую пару и выдать открытый ключ 1 – выдать ранее сгенерированный открытый ключ 2 – сгенерировать новую ключевую пару и ничего не выдавать
P2	00 – команда применяется к текущему файлу >0 - идентификатор ключа
Lc	размер данных
данные	инкапсулированные случайные числа или ничего
Le	размер выходных данных
ответ	открытый ключ или ничего

Для улучшения качества генерируемой ключевой пары, снаружи может передаваться случайная последовательность, инкапсулированная в шаблон KAT (key agreement template) с тэгом A6 в объекте 94 (random number). Допускается передача шаблона KAT с единственным объектом 94 нулевой длины – это эквивалентно отсутствию входных данных. Использование во входных данных шаблонов отличных от KAT или шаблона KAT с объектами отличными от 94-го не допускается.

В ответе на команду открытый ключ передается во внутреннем формате карты, зависящем от алгоритма ключа.

В настоящей версии ОС данной командой можно сгенерировать только ключ ГОСТ Р34.10-2001.

9.7.3 PERFORM SECURITY OPERATION

Команда PERFORM SECURITY OPERATION (PSO), фактически, объединяет несколько подкоманд, отвечающих за выполнение различных криптографических операций.

Все подкоманды имеют одинаковые байты CLA и INS в заголовке и отличаются только параметрами P1-P2.

Кодировка режимов команды PSO

Название операции	Действие	P1 (выход)	P2 (вход)
ENCIPHER (ENC)	Шифрование	86	80
DECIPHER (DEC)	Расшифрование	80	86
COMPUTE CRYPTOGRAPHIC CHECKSUM (CCC)	Вычисление имитовставки	8E	80
VERIFY CRYPTOGRAPHIC CHECKSUM (VCC)	Проверка имитовставки	00	A2
COMPUTE HASH (CH)	Вычисление хэша	90	80
COMPUTE DIGITAL SIGNATURE (CDS)	Вычисление ЭЦП	9E	9A
VERIFY DIGITAL SIGNATURE (VDS)	Проверка ЭЦП	00	A8
VERIFY CERTIFICATE (VC)	Проверка сертификата	00	BE|AE

Таблица 4: Операции команды PSO

Значения P1 и P2 могут принимать значения из таблицы «Таблица 5: Значения P1:P2 в команде PSO

Таблица 6: тип операции в зависимости от P1:P2

Параметры криптографической операции определяются активным SE. Если в SE не хватает данных для выполнения затребованной операции, то возникает ошибка.

Для файла SEF, содержащего текущий SE, должно быть разрешено ***использование***; для ключей, задействованных в криптографической операции, должно быть разрешено ***использование*** и в маске назначения ключа должно быть разрешено использование ключа для криптографических операций.

Если размер входных данных не позволяет передать их целиком в одной команде, то можно подать несколько команд одинакового типа подряд в режиме command chaining, который определяется по байту CLA (бит 5) в заголовке команды. Данная функциональность поддерживается для операций ENCIPHER, DECIPHER, COMPUTE CRYPTOGRAPHIC CHECKSUM, VERIFY CRYPTOGRAPHIC CHECKSUM и COMPUTE HASH. При этом для команды VERIFY CRYPTOGRAPHIC CHECKSUM длина объекта данных 80 указывается для текущей команды цепочки.

Особым случаем использования command chaining является вычисление хэша (COMPUTE HASH) с последующим вычислением/проверкой ЭЦП (COMPUTE/VERIFY DIGITAL SIGNATURE). В этом случае одна/несколько операций СН в режиме command chaining завершаются операцией CDS или VDS.

Также особый случай command chaining используется в операции VERIFY CERTIFICATE. Допустимо передать первой командой сертификат по шаблону BE, а второй завершающей командой – ЭЦП по шаблону AE.

В операциях зашифрования, расшифрования и вычисления/проверки имитовставки размер промежуточных открытых/зашифрованных данных должен быть кратен размеру блока соответствующего алгоритма. Промежуточными выходными данными для зашифрования/расшифрования является промежуточный результат операции, для вычисления/проверки имитовставки промежуточных выходных данных нет.

Общий формат команды PSO следующий:

тип команды	in/out (case 4)

формат команды:

CLA	00
INS	2A
P1	Тэг выходных данных или 00, если выходных данных нет
P2	Тэг входных данных или 00, если входных данных нет
Lc	размер входных данных
данные	входные данные (если есть)
Le	размер выходных данных
ответ	выходные данные (если есть)

Далее рассмотрим отдельные криптографические операции:

Вычисление имитовставки

(COMPUTE CRYPTOGRAPHIC CHECKSUM – CCC)

IN	
OUT	

Вычисление ЭП

(COMPUTE DIGITAL SIGNATURE – CDS)

IN	
OUT	

В качестве входных данных необходимо указать посчитанный хэш от данных, для которых вычисляется ЭЦП.

Допустимо использовать данную операцию последней командой в цепочке предшествующего вычисления хэша. В этом случае длина входных данных должна быть равно нулю.

Режим включения дополнительных данных из SE (auxiliary data) не поддерживается.

Вычисление хэша

(COMPUTE HASH – CH)

IN	
OUT	

Передача промежуточного хэша (по предыдущим блокам) не поддерживается, т.к. для хеширования длинных последовательностей используется режим command chaining.

Режим сохранения хэша в контексте не поддерживается.

Проверка имитовставки

(VERIFY CRYPTOGRAPHIC CHECKSUM – VCC)

IN	
OUT	

Проверка ЭП

(VERIFY DIGITAL SIGNATURE – VDS)

Проверка сертификата ключа

(VERIFY CERTIFICATE – VC)

Зашифрование

(ENCIPHER – ENC)

Расшифрование

(DECIPHER – DEC)

9.8 Команды аутентификации

Команды данной группы включают команды криптографической внешней и внутренней аутентификации, команды предъявления пароля и управления данными, связанными с аутентификацией.

Во всех командах данной группы приняты следующие соглашения о кодировании параметров (если явно не оговорено иное):

- P1=0
- Если P2=0, то параметры берутся из текущего SE, иначе P2 – ссылка на ключ.

Ссылка на ключ содержит идентификатор санкции для аутентификации, ассоциированный с ключом и принимает значение в диапазон ***1..127***. Старшие идентификаторы санкций зарезервированы для использования кодом ОС и расширений, а ***0*** не должен использоваться, т.к. P2=0 имеет специальный смысл. В целях повышения совместимости с ISO 7816-4 рекомендуется использовать идентификаторы санкций лишь в диапазоне ***1..31***.

Поиск ключа осуществляется в текущем DF и выше (до MF).

типы файлов:

DF (MF)	BF	FRF	VRF	CRF	TF	KF	ARF
-	-	-	-	-	-	+	-

9.8.1 GET CHALLENGE

Команда генерирует и возвращает случайное число указанного размера. Число запоминается в текущем SE и может быть использовано в последующих командах.

тип команды	out (case 2)

формат команды:

CLA	00
INS	84
P1	00
P2	00
Lc	—
данные	—
Le	размер запрошенного числа 1..32
ответ	случайное число

», определяющие тип входных и выходных данных. Тип операции определяется на основании таблицы «Таблица 6: тип операции в зависимости от P1:P2», исходя из типов входных и выходных данных.

Тэги операций в команде PSO

Тэг	Тип данных
	открытые (исходные) данные
	байт выравнивания, за которым идет криптограмма по произвольным данным
	имитовставка
	хэш
	данные для вычисления ЭЦП
	ЭЦП
	шаблон для проверки имитовставки
	шаблон для проверки ЭЦП
	не самоописываемый сертификат
	самоописываемый сертификат

Таблица 5: Значения P1:P2 в команде PSO

Таблица 6: тип операции в зависимости от P1:P2

Параметры криптографической операции определяются активным SE. Если в SE не хватает данных для выполнения затребованной операции, то возникает ошибка.

Для файла SEF, содержащего текущий SE, должно быть разрешено ***использование***; для ключей, задействованных в криптографической операции, должно быть разрешено ***использование*** и в маске назначения ключа должно быть разрешено использование ключа для криптографических операций.

Если размер входных данных не позволяет передать их целиком в одной команде, то можно подать несколько команд одинакового типа подряд в режиме command chaining, который определяется по байту CLA (бит 5) в заголовке команды. Данная функциональность поддерживается для операций ENCIPHER, DECIPHER, COMPUTE CRYPTOGRAPHIC CHECKSUM, VERIFY CRYPTOGRAPHIC CHECKSUM и COMPUTE HASH. При этом для команды VERIFY CRYPTOGRAPHIC CHECKSUM длина объекта данных 80 указывается для текущей команды цепочки.

Особым случаем использования command chaining является вычисление хэша (COMPUTE HASH) с последующим вычислением/проверкой ЭЦП (COMPUTE/VERIFY DIGITAL SIGNATURE). В этом случае одна/несколько операций CH в режиме command chaining завершаются операцией CDS или VDS.

Также особый случай command chaining используется в операции VERIFY CERTIFICATE. Допустимо передать первой командой сертификат по шаблону BE, а второй завершающей командой – ЭЦП по шаблону AE.

В операциях зашифрования, расшифрования и вычисления/проверки имитовставки

размер промежуточных открытых/зашифрованных данных должен быть кратен размеру блока соответствующего алгоритма. Промежуточными выходными данными для зашифрования/расшифрования является промежуточный результат операции, для вычисления/проверки имитовставки промежуточных выходных данных нет.

Общий формат команды PSO следующий:

тип команды	in/out (case 4)

формат команды:

CLA	00
INS	2A
P1	Тэг выходных данных или 00, если выходных данных нет
P2	Тэг входных данных или 00, если входных данных нет
Lc	размер входных данных
данные	входные данные (если есть)
Le	размер выходных данных
ответ	выходные данные (если есть)

Далее рассмотрим отдельные криптографические операции[17]:

Вычисление имитовставки

(COMPUTE CRYPTOGRAPHIC CHECKSUM – CCC)

IN	
OUT	

Вычисление ЭП

(COMPUTE DIGITAL SIGNATURE – CDS)

IN	
OUT	

В качестве входных данных необходимо указать посчитанный хэш от данных, для которых вычисляется ЭЦП.

Допустимо использовать данную операцию последней командой в цепочке предшествующего вычисления хэша. В этом случае длина входных данных должна быть равно нулю.

Режим включения дополнительных данных из SE (auxiliary data) не поддерживается.

Вычисление хэша

(COMPUTE HASH – CH)

[17] Операции приведены в том же порядке, что и в ISO 7816-8

IN	
OUT	

Передача промежуточного хэша (по предыдущим блокам) не поддерживается, т.к. для хеширования длинных последовательностей используется режим command chaining.

Режим сохранения хэша в контексте не поддерживается.

Проверка имитовставки

(VERIFY CRYPTOGRAPHIC CHECKSUM – VCC)

IN	
OUT	

Проверка ЭП

(VERIFY DIGITAL SIGNATURE – VDS)

Проверка сертификата ключа

(VERIFY CERTIFICATE – VC)

Зашифрование

(ENCIPHER – ENC)

Расшифрование

(DECIPHER – DEC)

9.9 Команды аутентификации

Команды данной группы включают команды криптографической внешней и внутренней аутентификации, команды предъявления пароля и управления данными, связанными с аутентификацией.

Во всех командах данной группы приняты следующие соглашения о кодировании параметров (если явно не оговорено иное):

- P1=0
- Если P2=0, то параметры берутся из текущего SE, иначе P2 – ссылка на ключ.

Ссылка на ключ содержит идентификатор санкции для аутентификации, ассоциированный с ключом и принимает значение в диапазон ***1..127***. Старшие идентификаторы санкций зарезервированы для использования кодом ОС и расширений, а ***0*** не должен использоваться, т.к. P2=0 имеет специальный смысл. В целях повышения совместимости с ISO 7816-4 рекомендуется использовать идентификаторы санкций лишь в диапазоне ***1..31***.

Поиск ключа осуществляется в текущем DF и выше (до MF).

типы файлов:

DF (MF)	BF	FRF	VRF	CRF	TF	KF	ARF
-	-	-	-	-	-	+	-

9.9.1 GET CHALLENGE

Команда генерирует и возвращает случайное число указанного размера. Число запоминается в текущем SE и может быть использовано в последующих командах.

тип команды	out (case 2)

формат команды:

CLA	00
INS	84
P1	00
P2	00
Lc	—
данные	—
Le	размер запрошенного числа 1..32
ответ	случайное число

9.9.2 INTERNAL AUTHENTICATE

Внутренняя аутентификация – доказательство внешнему миру наличия на карте определенного секрета.

Для блочных симметричных алгоритмов размер СЧ должен быть равен размеру блока, для асимметричных алгоритмов размер СЧ должен быть от 8 до 32 байт.

тип команды	in/out (case 4)

формат команды:

CLA	00
INS	88
P1	00
P2	ссылка на ключ
Lc	размер входных данных (обычно 8)
данные	challenge – случайное число
Le	размер выходных данных (обычно 6)
ответ	криптограмма

Размер входных и выходных данных определяется свойствами ключа и текущего SE.
По умолчанию, размер challenge – 8 байт, размер ответной криптограммы – 6 байт.

9.9.3 EXTERNAL AUTHENTICATE

Внешняя аутентификация – проверка картой, что внешняя сторона владеет определенным секретом. Команда EXTERNAL AUTHENTICATE должна следовать за командой GET CHALLENGE, причем размер запрошенного случайного числа должен соответствовать алгоритму ключа и/или текущим настройкам SE.

Для блочных симметричных алгоритмов размер СЧ должен быть равен размеру блока, для асимметричных алгоритмов размер СЧ должен быть от 8 до 32 байт.

тип команды	in/out (case 4)

формат команды:

CLA	00
INS	82
P1	00
P2	ссылка на ключ
Lc	размер входных данных (6)
данные	криптограмма
Le	–
ответ	–

9.9.4 GENERAL AUTHENTICATE

Обобщенная команда взаимной аутентификации объединяет функциональность ряда протоколов взаимной аутентификации в едином синтаксисе.

Последовательность команд GENERAL AUTHENTICATE, реализующая единый протокол взаимной аутентификации, объединяется в цепочку (command chaining).

Входные и выходные данные передаются внутри шаблона ***7C*** – данные для динамической аутентификации (только в случае, если данные наличествуют; если данных нет, объемлющий шаблон не используется). Объекты данных внутри этого шаблона имеют контекстно-зависимые тэги, перечисленные в следующей таблице:

Тэг	**Значение**
	Witness – синхропосылка (RFU - не используется)
	Challenge – случайное число
	Response – криптограмма
	Committed challenge – хэш раннее использованных СЧ (RFU - не используется)
	Authentication code – хэш данных операции (RFU - не используется)
	Exponential – компонент в алгоритме DH (RFU - не используется)
	Session Key (RFU - не используется)
	ЭЦП компоненты ключа (RFU - не используется)

Если атрибуты ключа, используемого в данной операции, требуют генерации сессионного ключа, то автоматически генерируется и применяется сессионный ключ.

тип команды	in/out (case 4)

формат команды:

CLA	00
INS	87
P1	00
P2	00
Lc	размер входных данных
данные	входные данные
Le	размер выходных данных
ответ	выходные данные

INS=86 не поддерживается.

Выбор режима команды определяется входными данными (тэгом и длиной) и предыдущими командами в цепочке.

Предусмотрены следующие стандартные протоколы использования команды:

Взаимная аутентификация по симметричной схеме

Данный вариант взаимной аутентификации симметричен, в том смысле, что ответной частью при обмене может быть аналогичная карта (модуль безопасности).

В ходе взаимной аутентификации вырабатывается сессионный ключ (если у ключа есть соответствующие свойства).

Недостатком данного варианта является то, что собственная криптограмма выдается

до проверки криптограммы ответной части, что является непременным требованием при реализации симметричной схемы.

В настройках SE указывается в каком качестве: модуля безопасности или клиентской карты – выступает карта в настоящий момент.

блок данных	назначение блока данных	тэг и длина данных
Request 1	запрос СЧ	81 00
Response 1	получить СЧ	81 L(16)
Request 2	передать СЧ ответной части	81 L(16); 82 00
Response 2	получить криптограмму	82 L(6)
Request 3	передать криптограмму ответной части	82 L(6)
Response 3	—	—

Взаимная аутентификация по несимметричной схеме.

Данный вариант взаимной аутентификации не симметричен, в том смысле, что ответной частью при обмене не может быть аналогичная карта (как и в MUTUAL AUTHENTICATE).

В ходе взаимной аутентификации вырабатывается сессионный ключ (если у ключа есть соответствующие свойства).

Преимуществом данного варианта является то, что собственная криптограмма выдается только в случае успешной проверки криптограммы ответной части.

блок данных	назначение блока данных	тэг и длина данных
Request 1	передать СЧ ответной части	
Response 1	получить СЧ карты	
Request 2	передать криптограмму ответной части	
Response 2	получить криптограмму карты	

9.9.5 VERIFY

Команда проверки пароля.

тип команды	in (case 3)

формат команды:

CLA	00
INS	20
P1	00
P2	ссылка на ключ
Lc	размер входных данных
данные	пароль или ничего (для получения значения счетчика попыток)
Le	—
ответ	—

характерные статусы ошибок:

63 CX	оставшееся число попыток при неверном предъявлении
69 83	пароль блокирован

Если входные данные отсутствуют, то карта сразу возвращает 9000, если пароль уже был предъявлен, или оставшееся число попыток предъявления.

INS=21 не поддерживается.

9.9.6 CHANGE REFERENCE DATA

Загрузить ключ или пароль.

Если файл был пуст, то требуется право доступа Put, если происходит смена ключа, то требуется право доступа Change.

Загрузка секретного ключа RSA выполняется в режиме chaining: сначала выполняется загрузка открытого ключа (e, n) с CLA=10, затем загрузка секретного ключа (d) с CLA=00.

тип команды	in (case 3)

формат команды:

CLA	10 — загрузка открытого ключа в составе всей ключевой пары RSA 00 – все остальные варианты
INS	24
P1	01
P2	P2=0 – применить команду к текущему файлу P2>0 - ссылка на ключ
Lc	размер входных данных: 32 байта — ГОСТ 28147-89 8 байт — DES 16 байт – 3DES 64 и 32 байта — открытый (x, y) и секретный ключ ГОСТ 34,10-2001 136 и 128 байт — открытый (e, n) и секретный (d) ключ RSA
данные	ключ
Le	–
ответ	–

9.9.7 RESET RETRY COUNTER

Восстановление счетчика оставшихся попыток до его максимального значения и разблокирование ключа (пароля).

тип команды	case 1

формат команды:

CLA	00
INS	2C
P1	03
P2	ссылка на ключ
Lc	—
данные	—
Le	–
ответ	–

Режимы P1=00,01,02 (связанные с передачей кода разблокирования и сменой пароля) не поддерживаются.

9.10 Сервисные команды

Сервисные команды предназначены для контроля целостности карты и диагностики сбоев.

9.10.1 VALIDATE CARD

Команда VALIDATE CARD выполняет расширенную диагностику карты
Команда работает на всех фазах жизни карты кроме блокирования.

При отсутствии возможности выдать код ошибки непосредственно в ответе на какую-либо команду (в том числе в случаях, определяющих переход карты в состояние “MUTE”), ОС сохраняет на карте код ошибки. Этот код впоследствии может быть считан с помощью данной команды.

Если же протокол обработки ошибки не позволяет ОС сохранить этот код, он может быть получен с помощью «теплого» сброса карты (т.е. Reset без снятия питания). В этом случае карта выдаст диагностический ATR, последние два байта которого будут содержать код ошибки.

тип команды	out (case 2)

формат команды:

CLA	
INS	
P1	0 – получить информацию об ОС и микроконтроллере 1 – выдать флаги диагностики карты 2 – проверить EEPROM 3 – выполнить расширенную диагностику оборудования 4 – получить случайные числа со встроенного ДСЧ 5 – проверить криптографические алгоритмы на контрольной задаче 6 – получить OTP производителя микроконтроллера 7 – вычислить криптографическую контрольную сумму ОС или отдельных модулей ОС 8 – прочитать криптограмму функциональности 9 – вычислить арифметическую контрольную сумму ОС или отдельных модулей ОС A – получить сохраненные регистры статуса B – получить сохраненный код ошибки (“MUTE”) и количество ее повторов
P2	P1=4 – количество случайных байт N P1=7,9 0 – контрольная сумма ОС 1 – контрольная сумма криптографического модуля 2 – контрольная сумма модуля ФКН P1=9 – номер ошибки (0..2, где 0 – последняя)
Lc	P1=7 – 32 Для остальных P1 – 0
данные	P1=7 – ключ для криптографического алгоритма ГОСТ 28147-89 Для остальных P1 – данные отсутствуют
Le	размер данных
ответ	<table><tr><td>тэг</td><td>длина</td><td>значение</td></tr></table>

P1=0 (Le=31)		
смещение	размер	значение
P1=1 (Le=1)		
P1=2 (Le=8)		
P1=3 (Le=2)		
0	2	Флаги контроля оборудования.
P1=4 (Le=N)		
0	N	N байт случайных чисел с ДСЧ
P1=5 (Le=2)		
P1=6 (Le=64)		
P1=7 (Le=4)		
0	4	
P1=8 (Le=5)		
0	5	
P1=9 (Le=4)		
0	4	
P1=A (Le=5)		
0	5	
P1=B (Le=3)		
0	2	
2	1	

9.10.2HANG CARD

Проверить срабатывание защитных механизмов карты.
После выполнения команды карта отключается и для ее активизации требуется Reset.
Если команда вернет какой бы то ни было либо статус, значит произошла ошибка.

В параметре P1 указывается тип тестируемого защитного механизма.
Для ST23:

- 1 – переход на несуществующий адрес
- 2 – обращение по несуществующему адресу
- 3 – неверный код команды
- 6 – программная защита
- 7 – проверка защиты памяти

тип команды	case 1

формат команды:

CLA	
INS	
P1	
P2	
Lc	
данные	
Le	
ответ	

9.11 Справочник по командам

Команды фазы эксплуатации в алфавитном порядке

Команда ОС	CLA	INS	Раздел
ACTIVATE FILE	00	44	9.1.2
APPEND RECORD	00	E2	9.5.1
CHANGE REFERENCE DATA	00	24	9.9.6
CREATE FILE	00	E0	9.1.1
DEACTIVATE FILE	00	04	9.1.3
DELETE FILE	00	E4	9.1.4
EXTERNAL AUTHENTICATE	00	82	9.9.3
GENERAL AUTHENTICATE	00	87	9.9.4
GENERATE KEY PAIR	00	46	9.7.2
GET CHALLENGE	00	84	9.9.1
GET DATA	00	CA/CB	9.3.1
INTERNAL AUTHENTICATE	00	88	9.9.2
MANAGE SECURITY ENVIRONMENT	00	22	9.7.1
PERFORM SECURITY OPERATION	00	2A	9.7.3
PUT DATA	00	DA/DB	9.3.2
READ BINARY	00	B0	9.4.1
READ RECORD	00	B2	9.5.2
RESET RETRY COUNTER	00	2C	9.9.7
SELECT	00	A4	9.2.1
TERMINATE CARD USAGE	00	FE	9.1.5
UPDATE BINARY	00	D6	9.4.2
UPDATE RECORD	00	DC	9.5.3
VALIDATE CARD	80	FE	9.10.1
VERIFY	00	20	9.9.5
WRITE BINARY	00	D0	9.4.3

9.12 Справочник по статусам команд

Статусы выполнения команд

Статус	Расшифровка
65 81	Ошибка целостности EEPROM
67 00	Неверная длина данных
67 01	Запрошено слишком много данных
67 02	Слишком много данных для проведения SM
67 10	Значение Le превышает размер коммуникационного буфера
67 11	Значение Lc превышает размер коммуникационного буфера
69 00	Выполнение команды запрещено
69 10	Директория не пуста (при попытке удалить)
69 11	Нельзя удалять MF
69 81	Команда не совместима со структурой файла
69 82	Доступ запрещен
69 83	Ключ блокирован
69 85	Не выполнены предварительные условия
69 86	Команда не применима к данному типу файла
69 87	Отсутствуют данные необходимые для проведения SM
69 88	Некорректный объект SM
69 89	Команда недопустима на данной фазе жизни файла или карты
69 90	Ошибочная имитовставка при SM
69 91	Поврежден файл
69 92	Правило доступа с указанным номером уже определено
69 93	Требуется генерация производного ключа
69 94	Ключ не предназначен для выполнения указанной операции
69 95	Криптографическая операция не может быть продолжена
69 96	Неверная ЭЦП
69 97	Необходимо использовать SM
69 98	Запрещено менять настройки SM при активном SM
69 99	Ключ не найден
69 9A	Не найден указанный CRT
69 9B	Недопустимый криптографический алгоритм ключа
69 9C	Неверное выравнивание данных при SM
69 9D	Ключ не загружен
69 9E	Некорректное правило разграничения доступа
69 A1	Раздел уже занят
69 A2	Пересечение разделов
69 A3	Вектор расширения занят или блокирован
69 A4	Список команд-расширений уже полон
69 A5	Адрес обработчика вне заполненной части текущего раздела
69 A6	Карта еще не отформатирована
69 A7	Недостаточно памяти
69 A8	Вектор вне допустимого диапазона
69 A9	Команда вне допустимого диапазона
6A 00	Неверные параметры команды
6A 80	Неверные данные команды
6A 82	Файл не найден Ошибка создания файла

6A 83	Запись не найдена
6A 84	Не хватает места
6A 86	Неверные параметры P1-P2
6A 91	Неверные параметры криптографического алгоритма
6A 92	Неправильные параметры операции
6A 93	Недопустимая длина СЧ
6A 94	Недопустимый номер санкции
6A 95	Несовместимость алгоритма и назначения ключа или битов назначения ключа
6B 00	Указано смещение за границами файла
6C xx	Неверное значение Le; xx – правильное значение Le
6D 00	Не найдена команда с данным INS
6E 00	Не найдена команда с данным CLA
91 00	Аппаратная ошибка
91 01	Таймаут ожидания данных
91 02	Обращение по несуществующему адресу в EEPROM
91 03	Переполнение буфера транзакций
91 04	Недостаточно оперативной памяти
91 05	Ошибочное освобождение памяти
91 06	Commit без открытия транзакции
91 07	Ошибка в транзакционном буфере
91 08	Недостаточно место в файловой системе
91 09	Повреждены служебные области EEPROM
91 0A	Поврежден EEPROM
91 0B	Недопустимые условия окружающей среды
91 0C	Не найден идентификатор памяти
91 0D	Неправильные данные для записи
91 0E	Сброс из-за ошибки оборудования
91 10	Не пройден контроль конфигурации оборудования
91 11	Ошибка записи EEPROM
91 12	Ошибка чтения EEPROM
91 13	Неправильная фаза жизни
91 14	Полный отказ ДСЧ
91 15	Статистический отказ ДСЧ
91 16	СЧ не подходит для ГОСТ Р34.10-2001
91 17	HANG CARD
91 18	Ошибка счетчика операций криптосопроцессора
91 19	Ошибка акселератора CRC
91 1A	Ошибка генератора
91 1B	Ошибка таймера
91 1C	Ошибка акселератора DES
91 1D	Ошибка криптосопроцессора
91 1E	Ошибка контрольной суммы ROM
91 20	Алгоритм не определен
91 21	Неверный параметр алгоритма
91 22	Операция неприменима к данному типу алгоритма
91 23	Режим не определен
91 24	Неверный размер входных данных
91 25	Недостаточный размер буфера для выходных данных
91 26	Ошибка при выполнении криптографического преобразования
91 27	Тэг не определен

91 28	Контекст не инициализирован или поврежден
91 29	Переполнение буфера контекстов
91 2A	Переполнение буфера данных
91 2B	Неверное направление криптооперации(зашифровать/расшифровать)
91 2C	Совпадение случайных чисел при выработке сессионного ключа
91 2D	Неверное значение индекса параметра по умолчанию
91 2E	Ошибка ДСЧ
91 2F	Ошибка выравнивания
91 30	Неправильные входные данные
91 31	Неправильный входной ключ
93 00	Ошибка в ОС
95 00	Функция не реализована

10 СТРУКТУРЫ ДАННЫХ НА КАРТЕ

10.1 FCI, FCP, FMD

FCP – File Control Parameters. Тэг: ***62***.

FCP содержит данные, приведенные в таблице «Тэги данных в FCP» (согласно ISO 7816-4).

FMD – File Management Data (tag ***64***) – множество элементов данных, характеризующих приложение (см. 5.3.3):

- Идентификатор приложения (AID)
- Название приложения (Application label)
- Другие объекты, характеризующие приложение

Фактически, FMD для DF – содержание контекста DF; для EF FMD не определен.

FCI – File Control Information (tag ***6F***) представляет собой конкатенацию элементов данных FCP и FMD.

Тэги данных в FCP

Тэг	Длина	Значение+	Тип файла	выбор/ создание
		Размер тела файла для KF и SEF - размер автоматически вычисляется на основе алгоритма	все кроме KF и SEF	+ / +
		Полный размер файла	все	+ / -
		Тип файла	DF, BF, VRF, TF, KF, ARF, SEF	+ / +
		Тип файла Способ кодирования данных = константе ***C1*** Размер записи	FRF, CRF	+ / +
		Идентификатор файла	все	+ / +
		Имя приложения (AID)	DF	? / ?
		атрибуты доступа	все	+ / +
		фаза жизни файла	все	+ / ?
		Дополнительная информация в формате BER-TLV (см.**Ошибка! Источник ссылки не найден.**)	все	? / ?

При создании файла тэг 8A (фаза жизни) может не указываться, в этом случае фаза жизни созданного файла – «инициализация». Если тэг присутствует, то он должен иметь значение «инициализация» или «использование»

Тэги нестандартных данных в FCP (шаблон A5)

Тэг	Длина	Значение	Тип файла	выбор/ создание
		Гарантированный размер DF	DF	? / ?
		Свободное место в DF	DF	? / -
		Максимальный размер создаваемого файла	DF	+ / -
		Требование вычисления CRC тела файла (по умолчанию не требуется)	все	+ / ?
		Максимальное количество записей (≤64)	ARF	+ / +
		Доступное количество шаблонов доступа (максимальное - при создании, текущее незанятое при выборе)	SEF	+ / +
		Алгоритм ключа (см.Файлы ключей KF п.5.3.1)	KF	+ / +
		Назнач. ключа (см.Файлы ключей KF п.5.3.1)	KF	+ / +
		Идентификатор санкции при предъявлении (0 – RFU)	KF	+ / +
		Идентификатор санкции при использовании SM (0 – санкция не устанавливается, может совпадать с элементом 87)	KF	+ / +
		Максимальное число попыток предъявления	KF	+ / +
		Маска разрешенных расширений в текущем DF (bit N соответствует разделу N) 00 – все расширения запрещены (по умолчанию) FF – все расширения разрешены	DF	+ / ?
		Признак наличия секретного ключа (по умолчанию 1/есть для асимметричного ключа и 0/нет для прочих)	KF	+ / ?
		Идентификатор SE (0 - RFU)	SEF	+ / +
		Уровень выдачи информации по SELECT для всех файлов в пределах DF: 0 – в полном объеме (по умолчанию) 1 - запрещена выдача правил разграничения доступа и специальных параметров KF 2 - запрещена выдача привил разграничения доступа, специальных параметров всех типов файлов 3 - выдается пустой FCP 4 - не выдается ни FCP, ни FMD, ни FCI	DF	+ / ?
		Настройка алгоритма. Для ГОСТ 28147-89, ГОСТ Р 34.11-94 – вариант узлов замен: 0 – по умолчанию (Oscar или из EEPROM) 1 – Oscar 1.1 и выше 2 – Oscar 1.0 3 – UniCOS 4 – стандартный (из «Инструкции по контролю») 5..9 – КриптоПро A..D, H Для ГОСТ Р 3410-2001 – вариант параметров алгоритма: 0 – по умолчанию (Oscar 2 или из EEPROM) 1 – Oscar 2 (совпадает с КриптоПро B) 2 – КриптоПро A 3 – КриптоПро C 4 – стандартные (из стандарта)	KF	+ / ?
		Маскирование ключа: 0 – выключено 1 – включено	KF	+ / ?

10.2 Фазы жизни файла

Фазы жизни файла

Фаза инициализации	03
Фаза использования	05
Фаза блокирования (deactivated)	04

10.3 Типы файлов

Типы файлов

DF (MF)	
BF	
FRF	
VRF	
CRF	
TF	
KF	
ARF	
SEF	

10.4 Способ кодирования данных

Согласно ISO 7816-4 (table 87) способ кодирования данных задается битовой маской `C1`:

- Поддерживаются TLV-файлы
- Команда Write выполняет операцию OR
- Единица адресации в бинарных файлах – байт (8 бит)
- Значение FF запрещено в первом байте тэгов BER-TLV записей.