SIM-КАРТЫ: ПРОТОКОЛЫ ОБМЕНА ДАННЫМИ

Большинство пользователей редко думают о том, что их мобильный телефон представляет собой замечательное вычислительное устройство, позволяющее им совершать телефонные звонки, обмениваться текстовыми сообщениями или играть в он-лайн игры. Модуль идентификации абонента (далее SIM- карта), в мобильном телефоне, фактически, является полноценным микрокомпьютером с операционной и файловой системой. Протоколы обмена данными, используемые между мобильным оборудованием или устройством чтения карт и SIM- картой, – это комбинация типов протоколов, которые должны быть знакомыми любому, кто изучал эталонную модель взаимодействия открытых систем ("Open Systems Interconnection", далее OSI), то есть, уровни управления передачей данных и прикладные уровни (хотя, модель T=0 смешивает уровни).

Эта статья пытается в общих чертах описать устройство этих протоколов и способ их использования.

Подача питания на карту

Когда в считывающее устройство или телефонную трубку вставлена смарт-карта, или, в нашем обсуждении, SIM- карта, питание на неё подаётся не сразу. Сначала выполняется проверка на предмет того, правильно ли вставлена карта, во избежание её разрушения.

После проверки правильного позиционирования карты устройство чтения карт ("card accepting device", далее CAD) или мобильное оборудование ("mobile equipment", далее ME) подаёт на карту рабочее напряжение, и карта переводится в режим ожидания. Затем по линии контакта (C2) на лицевой стороне SIM-карты, подаётся команда RST. Команда RST остаётся в состоянии с низким уровнем установленный период времени, а потом приводится в состояние с высоким уровнем. Это является сигналом для SIM- карты начать последовательность инициализации.

Последовательность инициализации заканчивается тем, что SIM-карта отправляет ответ на сброс ("Answer to Reset", далее ATR). Основная цель этого ответа – указать состояние последовательности подачи питания на смарт-карту. Он также передаёт информацию, которую требует считывающее устройство, чтобы оптимизировать скорость передачи данных между устройством и картой. Общая длина последовательности ATR ограничена 33 байтами (ATR должен придерживаться структуры, указанной стандартом ISO 7816-3). В таблице ниже показан ATR в виде, в котором он описан в ISO 7816-3.

ID символа	**Описание**
Стартовый символ	*Секция*
TS	Обязательный стартовый символ
Символ формата	*Секция*
T0	Указатель присутствия символов интерфейса
Символ интерфейса	*Секция*
TA_1	Глобальный, коды F1 и D1
TB_1	Глобальный, коды 11 и Pl1
TC_1	Глобальный, код N
TD_1	Коды Y_2 и T
TA_2	Специальный
TB_2	Глобальный, код Pl2
TC_2	Специальный
TD_2	Коды Y_3 и T
TA_3	TA_i, TB_i и TC_i – специальные

...TD_i	Коды Y_{i+1} и T
Символ «истории»	*Секция*
T1	Специальная информация карты
...TK	(Максимум 15 символов)
Контрольный символ	*Секция*
TCK_2	Необязательный контрольный символ

Таблица № 1: Структура ATR[1].

TS и T0 – единственные обязательные байты в последовательности ATR. Стартовый символ TS используется, чтобы установить условные обозначения для назначения битов и порядка битов. T0 используется, чтобы указать на присутствие или отсутствие последующих символов интерфейса или истории. Старшие 4 бита или полубайт (биты 5 – 8) обозначен как Y1 и сообщает о присутствии необязательных символов, основанных на логической единице (1) в следующих позициях двоичного разряда:

- Бит 5 указывает, что присутствует TA1;
- Бит 6 указывает, что присутствует TB1;
- Бит 7 указывает, что присутствует TC1;
- Бит 8 указывает, что присутствует TD1.

Младший полубайт (биты 1 – 4) обозначен как K и интерпретируется как числовое значение в диапазоне 0 – 15. Он указывает число присутствующих символов истории.

Символы интерфейса используются, чтобы выбрать протокол и параметры, применяемые для последующей связи в соответствии с протоколом более высокого уровня между смарт-картой и считывающим устройством. Символ интерфейса TA1 наиболее важен в этой статье, поскольку он предоставляет необходимую информацию, чтобы достичь оптимальной скорости связи между считывающим устройством и картой. Значение Y1 указывает, присутствуют ли TA_1, TB_1, TC_1 или TD_1, а значение Y_i (закодировано в TD_{i-1}) определяет присутствие TA_i, TB_i, TC_i и TD_i. Ниже объясняется назначение нескольких самых важных из этих символов.

- TA1: Кодирует FI в старшем полубайте (биты 5 – 8) и DI в младшем полубайте (биты 1 – 4). Они используются, чтобы определить коэффициент преобразования тактовых импульсов (F) и коэффициент регулирования скорости передачи битов (D) и максимальную поддерживаемую тактовую частоту согласно таблицам ниже. Если не указано другое, значения по умолчанию F и D: F = 372 и D = 1.

FI	0000	0001	0010	0011	0100	0101	0110	0111
F	Внутренний генератор тактовых импульсов	372	558	744	1116	1488	1860	RFU*
***f* max (МГц)**	-	5	6	8	12	16	20	-

FI	1000	1001	1010	1011	1100	1101	1110	1111
F	RFU*	512	768	1024	1536	2048	RFU*	RFU*
***f* max (МГц)**	-	5	7,5	10	15	20	-	-

Таблица №2: Значение старших четырёх битов байта TA1 команды ATR[1].
где, *RFU (Reserved For Future Use) – Зарезервировано для использования в будущем.

DI	0000	0001	0010	0011	0100	0101	0110	0111
D	RFU*	1	2	4	8	16	32	RFU*

DI	1000	1001	1010	1011	1100	1101	1110	1111
D	12	20	1/2	1/4	1/8	1/16	1/32	1/64

Таблица № 3: Значение младших четырёх битов байта TA1 команды ATR[1].
где, *RFU (Reserved For Future Use) – Зарезервировано для использования в будущем.

- TB_1, TB_2: Используются, чтобы закодировать информацию относительно программирующего напряжения и тока.
- TC_1: Интерпретируемый как 8- битовое целое число без знака, представляющее дополнительный защитный интервал, который требуется между символами (N). Значение по умолчанию N равно 0.
- TD_1: Кодирует Y1 в старшем полубайте и тип протокола в младшем полубайте. ISO 7816-3 определяет два протокола:: протокол T=0 и протокол T=1. T=0 – асинхронный полудуплексный посимвольный протокол, в котором необходимо получить подтверждение для каждого отправленного байта. В противоположность, T=1 – асинхронный полудуплексный блочный протокол, в котором можно отправить ряд байтов перед подтверждением получения. Если TD_1 не присутствует, то T=0 используется неявно.

Символы «истории» обычно используются, чтобы указать тип, модель и применение определенной карты. Они обычно определяются изготовителем или эмитентом карты. Установленного стандарта для данных в этих битах «истории» не существует. Контрольный символ (TCK) используется, чтобы определить, произошла ли ошибка передачи при отправке ATR с карты на считывающее устройство. TCK – это контрольная сумма, вычисленная так, что, результат выполнения поразрядной операции «исключающее ИЛИ» на всех байтах в ATR от T0 до TCK равен нулю.

Транспортные протокольные блоки данных

Транспортные протокольные блоки данных (“Transmission Protocol Data Units”, далее TPDU) – это структуры данных, которые передаются в обоих направлениях между SIM-картой и CAD или ME. Два протокола, обычно используемые для этой связи, – это протокол T=0 и протокол T=1.

T=0

Этот протокол является протоколом с байтовой организацией, что означает, что минимальная единица информации, передаваемая через канал ввода-вывода, – это байт. Обработка ошибок заложено в сам протокол. Этот протокол имеет тенденцию смешивать элементы протокола канального уровня и протокола прикладного уровня.

TPDU протокола T=0 состоит из двух структур данных – одна отправляется со считывающего устройства или устройства на карту, а одна отправляется назад с карты (подобно команде и отклику). Заголовок команды протокола T=0 включает следующие пять полей.

- CLA – однобайтовое поле, которое является классом команд;

- INS – однобайтовое поле, которое определяет специальную команду из набора команд CLA;
- P1 – однобайтовое поле, которое используется для определения адресации, применяемой командами INS и CLA;
- P2 – однобайтовое поле, которое также используется в адресации;
- P3 – однобайтовое поле, которое используется для определения числа байтов, передаваемых на карту или с неё как часть выполнение команды[2].

В таблице ниже перечислены некоторые из команд, которые могут содержаться в байте CLA.

Байт CLA	**Набор команд**
0X	Команды ISO 7816-4 (файлы и безопасность)
с 10 по 7F	Зарезервировано для использования в будущем
8X или 9X	Команды ISO 7816-4
AX	Команды, характерные для приложения/ производителя
с B0 по CF	Команды ISO 7816-4
с D0 по FE	Команды, характерные для приложения/ производителя
FF	Зарезервировано для выбора типа протокола

Таблица № 4: Описания набора команд CLA[2].

В следующей таблице показаны наборы команд в байте INS для безопасности и доступа к файловой системе на карте.

Значение INS	**Название команды**	**Значение INS**	**Название команды**
0E	Стереть двоичные данные (Erase Binary)	C0	Получит отклик (Get Response)
20	Проверить (Verify)	C2	Конверт (Envelope)
70	Управлять каналом (Manage Channel)	CA	Получить данные (Get Data)
82	Внешняя аутентификация (External Authenticate)	D0	Сделать запись в двоичных данных (Write Binary)
84	Получить вызов (Get Challenge)	D2	Сохранить запись (Write Record)
88	Внутренняя аутентификация (Internal Authenticate)	D6	Обновить двоичные данные (Update Binary)
A4	Выбрать файл (Select File)	DA	Записать данные (Put Data)
B0	Прочитать двоичные данные (Read Binary)	DC	Обновить запись (Update Record)
B2	Прочитать запись(-и) (Read Record(s))	E2	Добавить запись (Append Record)

Таблица № 5: Коды INS в соответствии с ISO 7816-4[2].

Как было упомянуто ранее, протокол T=0 имеет тенденцию смешивать элементы с других уровней в модели OSI. Хотя параметры P1 и P2 определены на уровне протокола канального уровня, на самом деле они зависят от заданных команд. Другими словами, они зависят от информации протокола уровня приложения. Эти два параметра обеспечивают контроль и адресацию для команд, связанных с приложением, например команда **Select File**

(«Выбрать файл»), которая включает в себя выбор определённого файла в файловой системе, что затем позволяет выполнение других операций таких, как запись или чтение.

P3 также является параметром прикладного уровня. P3 обычно определяет число байтов, которые необходимо передать в течение выполнения команды, указанной INS. Данные либо отправляются с карты в считывающее устройство, либо поступают со считывающего устройства на карту.

После каждой команды TPDU, возвращается отклик TPDU. TPDU состоит из ряда байтов процедуры. TPDU содержит три обязательных поля и одно необязательное.

- ACK: указывает, что карта получила команду [CLA, INS];
- NULL: используется картой для управления обменом данных на канале ввода-вывода. Оно подаёт сигнал (считывающему устройству) о том, что карта всё ещё обрабатывает команду и поэтому считывающее устройство должно подождать, прежде чем посылать следующую команду;
- SW1: отклик статуса текущей команды;
- SW2: (необязательное) также передаёт отклик статуса на считывающее устройство[2].

Байт ACK – это повторение байта INS, отправленного командой TPDU. Второй байт NULL является способом отметки периода времени для обработки команды. Если за время периода тайм-аута отклик не получен, считывающее устройство может отправить последовательность RST, чтобы повторно инициализировать протокол. Как минимум один отклик NULL, полученный считывающим устройством, предотвратит это.

SW1 – байт состояния, отправляемый с карты на считывающее устройство, который сообщает результаты посланной команды; в некоторых командах карта может вернуть байты данных считывающему устройству или оборудованию. В этом случае байт состояния SW1 возвращается. Это является триггером для считывающего устройства выполнять следующую команду – команду **GetResponse**, которая фактически возвращает данные из ранее выполняемой команды. В протоколе T=0 с контролем чётности выполняется проверка ошибок. Для каждого переданного байта должно использоваться 11 битов. Бит чётности очищается или устанавливается так, чтобы сделать чётным общее число множества битов. Принимающая сторона канала может посмотреть на биты, переданные до бита чётности, и определить, какое значение чётности ожидать. Если фактическая передача не соответствует тому, что ожидалось, можно предположить, что произошла ошибка, и поэтому должна быть выполнена процедура восстановления. Это восстановление инициализируется принимающей стороной и включает в себя повторную передачу байтов, полученных по ошибке

T=1

Этот протокол является блочно-ориентированным, что означает, что определённый набор данных или блок пересылается как одно целое между картой и считывающим устройством. Этот блок данных может содержать пакет данных протокола прикладного уровня (“Application Protocol Data Unit”, далее APDU), определённый для отдельного приложения. Это является хорошим примером разбиения на уровни протоколов канального и прикладного уровней. Пересылка информации в виде единого блока требует передачи данных без ошибок. Процедура обнаружения и исправления ошибок намного сложнее для протокола T=1, чем для протокола T=0.

Это обнаружение ошибок выполняется при помощи символа продольного контроля (LRC) – немного более сложная проверка типа чётности, которая существует в протоколе T=0, или используя контроль с помощью циклического избыточного кода (CRC). Специальный используемый алгоритм CRC определён в стандарте ISO 3309.

Протокол T=1 использует три типа блоков:

1. Информационный блок: Используется для обмена данными между прикладным ПО карты и прикладным ПО на стороне считывающего устройства канала;
2. Блок готовности к приёму: Используется для положительного или отрицательного подтверждения на любом конце канала. Положительное подтверждение указывает, что блок данных был принят без ошибок. И наоборот, отрицательное подтверждение указывает, что во время получения блока данных была обнаружена ошибка;
3. Контрольный блок: Используется для передачи контрольной информации между считывающим устройством и картой.

Каждый из блоков данных имеет одинаковую структуру, которая состоит из следующих полей.

- Поле «пролога»: Обязательное поле размером три байта, включающее в себя следующие три элемента:
  - NAD: Адрес узла – используется для определения адресов источника и предполагаемого назначения блока;
  - PCB: Байт управления протоколом – используется для обозначения типа блока (информационный, готовности к приёму или контрольный);
  - LEN: Длина блока;
- Информационное поле: Необязательное поле, которое может быть длиной до 254 байтов и может содержать APDU;
- Поле «эпилога»: Обязательное поле длиной 1 или 2 байта, которое используется для обнаружения ошибок.

Элемент NAD также содержит два подполя: SAD – адрес источника, который показан тремя младшими битами NAD, и DAD – адрес назначения, который показан в битах, с пятого по седьмой, NAD.

В PCB два самых важных (старших) бита байта обозначают тип блока:

- Старший разряд установленный на ноль указывает на информационный бок;
- Два старших бита, установленных на единицу, указывают на контрольный блок;
- Два старших бита, один из которых установлен на единицу, а другой на ноль, указывают на блок готовности к приёму.

Пакеты данных протокола прикладного уровня

В стандарте ISO 7816-4 обозначены две области функционирования прикладного ПО:

- Файловая система: Набор функций предоставлен в виде интерфейса прикладного программирования (API). Используя прикладное ПО API на стороне считывающего устройства можно получить доступ к файлам в файловой системе;
- Функции безопасности: Можно использовать для ограничения доступа к прикладному ПО или к файлам карты.

Протоколы T=0 или T=1 используются для поддержки протоколов прикладного уровня между приложением смарт-карты и приложением считывающего устройства. Эти протоколы прикладного уровня обмениваются структурами данных, которые называются пакеты данных протокола прикладного уровня (APDU). Эта структура показана на следующей иллюстрации:

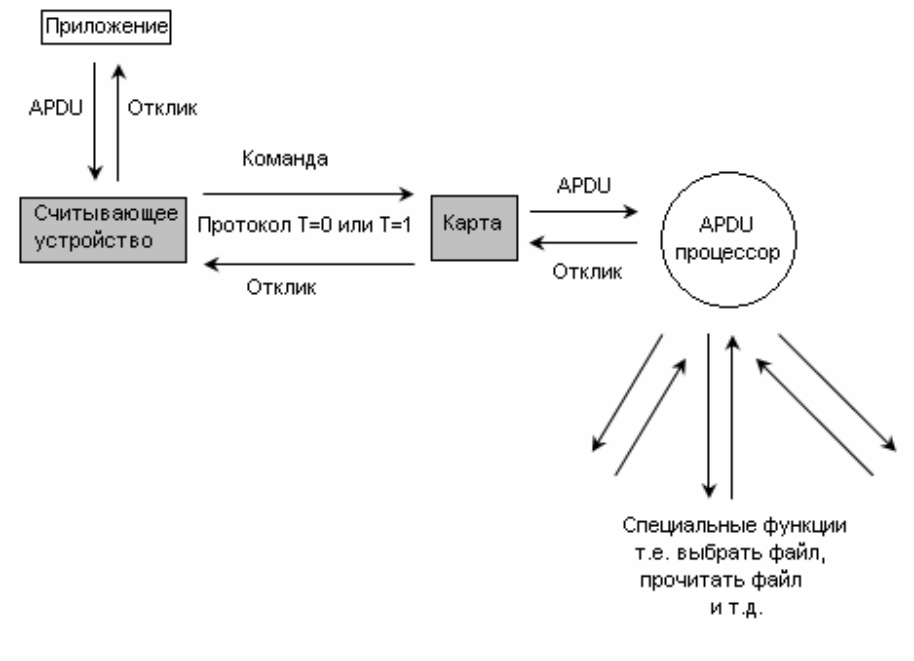

Иллюстрация 1: Структура связи приложений[2].

Структура APDU, определённая стандартом ISO 7816-4, очень похожа на структуру TPDU, которая используется в протоколе T=0. Фактически, при передаче APDU протоколом T=0, элементы APDU непосредственно накладываются на элементы TPDU.

В соответствии с ISO 7816, APDU является независимым протоколом канального уровня, который также определён на прикладном уровне.

Структура APDU

Существует два типа сообщений, используемых для поддержки протоколов прикладного уровня, в соответствии с ISO 7816-4: *командный APDU* (отправляемый из считывающего устройства на карту) и *APDU отклика* (отправляемый с карты на считывающее устройство).

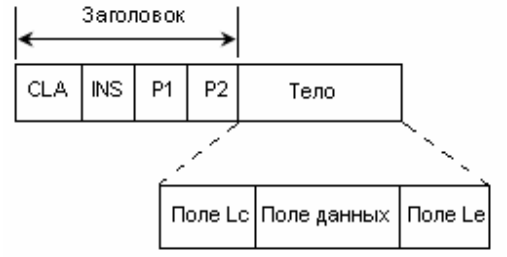

Иллюстрация 2: Структура командного APDU[2].

Командный APDU состоит из заголовка и тела (как показано выше). Заголовок включает в себя поля CLA, INS, P1 и P2. Как и в протоколе T=0, CLA и INS определяют класс приложения и команду.

P1 и P2 используются для уточнения специальных команд и им даётся специальное определение каждой командой [CLA, INS].

Тело APDU может изменяться по размеру и используется для передачи данных на процессор APDU карты как часть команды или для передачи отклика с карты на

считывающее устройство. Поле Lc указывает число байтов для передачи на карту как часть длины команды поля данных. Поле данных содержит информацию, которую необходимо отправить на карту, чтобы позволить её процессору APDU выполнить команду, указанную в APDU. Поле Le определяет число байтов, которые будут возвращены в считывающее устройство в APDU отклика.

Тело APDU может иметь 4 формы:

1. Данные не передаются на карту или с неё, поэтому APDU содержит только заголовок;
2. Данные не передаются на карту, но возвращаются с карты. Тело APDU содержит только непустое поле Le;
3. Данные передаются на карту, но не возвращаются с неё. Тело APDU включает поле Lc и поле данных;
4. Данные передаются на карту, а также возвращаются с неё как результат команды. Тело APDU включает поле Lc, поле данных и поле Le.

Иллюстрация 3: Структура APDU отклика.

APDU отклика состоит из заголовка и статусных байтов – более простая структура, чем у командного APDU. Тело либо пустое, либо включает поле данных – это зависит от определённой команды и её успешного или неуспешного выполнения процессором APDU. Если APDU содержит поле данных, его длина определяется полем Le в соответствующем командном APDU.

Статусные байты - состоят из двух полей информации о состоянии, которые называются SW1 и SW2. Эти поля возвращают код состояния, в котором один байт используется для указания категории ошибок, а другой используется для указания состояния команды или индикации ошибок.

Коды ошибок соответствуют схеме нумерации ISO 7816, в которой один байт используется для определения категории ошибки, а второй показывает специфическую ошибку.

КОД ВОЗВРАТА						
ПРОЦЕСС ЗАВЕРШЁН				ПРОЦЕСС ПРЕРВАН		
Нормально	Предупреждение			Ошибка выполнения		Ошибка при проверке
61XX or 9000	62XX	63XX		64XX	65XX	67XX to 6FXX

Таблица № 6: Коды возврата ошибок (стандарт ISO 7816)

Наконец, байт CLA в ADPU имеет две заслуживающих внимания особенности:

- Два младших бита могут указывать на логический канал связи между процессором APDU карты и приложением считывающего устройства;
- Следующие два старших бита могут указывать на то, что между приложением считывающего устройства и APDU карты используется безопасный обмен сообщениями.

Цель этой статьи состояла в том, чтобы описать общие коммуникационные протоколы, которые SIM-карта мобильного телефона (унаследовав их от смарт-карты) использует для связи с устройством чтения карт или мобильным оборудованием. Были обсуждены протоколы TPDU T=0 и T=1, а также структура APDU. Экспертам рекомендуется изучить работы из списка цитированной литературы для получения более глубокой информации о протоколах обмена данными и командах SIM-карт.

СПИСОК ЦИТИРОВАННОЙ ЛИТЕРАТУРЫ

1. Shillington Nicole, Waker Travers, *The Design of a Smart Card Interface* , http://www.cs.uct.ac.za/Research/DNA/SOCS/projectpage.html

2. Guthery Scott B., Jurgensen Timothy M., *Smart Cards: The Developers Toolkit* , Prentice Hall 2002, 78-113

Источник: SIM_CARD_PROTOCOLS.pdf

Перевод:
Бочков Д.С.
Капинус О.В. (info@computer-forensics-lab.org)
Михайлов И.Ю. (info@computer-forensics-lab.org)