Протокол обмена с Устройствами электронными для хранения и учёта типа СК (электронными сейфами) через Ethernet.

Введение

Ethernet порт контроллера соответствует стандарту 100BASE-TX. Обмен происходит по протоколу TCP/IP. *Контроллер* является сервером, он «слушает» на порту 18018.

Контроллер одновременно обрабатывает только одно соединение. Это означает, что клиент, установивший соединение с *контроллером*, блокирует подключение других клиентов до разрыва соединения. Это может использоваться клиентом для того, чтобы передать несколько команд *контроллеру* как одну транзакцию. С другой стороны, клиент не должен слишком долго удерживать соединение, если предполагается совместная работа нескольких клиентов.

После установления соединения клиента с *контроллером* последний при отсутствии обменов более 3-х секунд посылает клиенту пустой пакет, состоящий только из маркера начала и маркера конца, и затем разрывает соединение.

Команды обозначенные как поддерживаемые версией прошивки 8.3.0 и выше – это команды для прибора СК01; с версией прошивки 8.0.0 и выше – это команды для приборов СК12. Остальные команды поддерживаются всеми приборами типа СК (СК24, СК12, СК01). В новых разработках для СК01 и СК12 предпочтительно использовать «новые» команды.

Шифрование

Весь обмен шифруется. Шифрование производится по алгоритму Blowfish в режиме CBC. Вектор инициализации равен нулю.

Ключ шифрования алгоритма имеет длину 16 байт. Ключ шифрования генерируется контроллером секции управления и может быть отображён на дисплее контроллера в соответствующем разделе меню. На дисплее ключ представлен в виде 4-х десятичных чисел, разделённых дефисом. Для получения первого байта ключа шифрования алгоритма необходимо взять младший байт двоичного представления первого десятичного числа, для второго байта – старший байт двоичного представления первого десятичного числа. Для получения третьего и четвёртого байтов ключа шифрования аналогично используется второе десятичное число. Также получаем с пятого по восьмой байты ключа из третьего и четвёртого десятичных чисел. Оставшиеся восемь байт ключа получаем копированием первых восьми.

Шифромашина инициализируется при установлении соединения между клиентом и *контроллером*. После инициализации в шифромашину передаётся случайная последовательность – «затравка». Длина «затравки» – 4 байта. Шифромашина не переинициализируется до разрыва соединения.

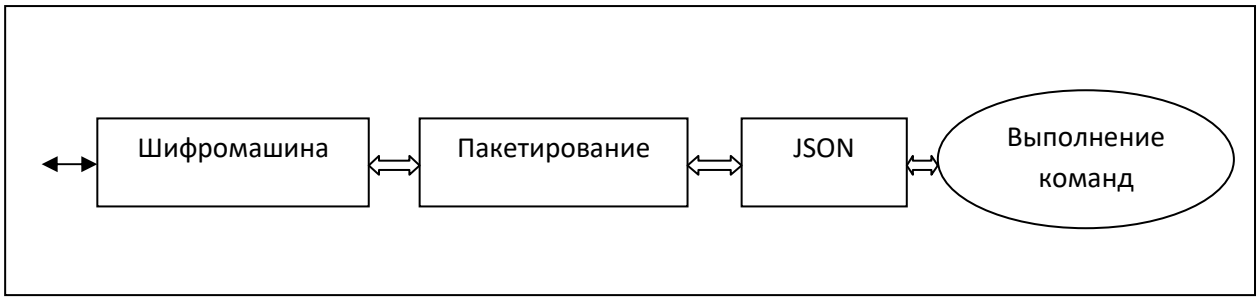
Пакеты

При установленном соединении обмен производится пакетами. За время одного соединения может быть передано произвольное количество пакетов как в одну, так и в другую сторону. Пакет начинается с *маркера начала* и заканчивается *маркером конца*. Между пакетами может быть произвольное количество *заполнителей*. Заполнители используются для того, чтобы шифромашина выдала последний зашифрованный блок без её закрытия.

Коды маркеров:

- *маркер начала* – 0x12
- *маркер конца* – 0x14
- *заполнитель* – 0x16

Контроллер обрабатывает пакеты последовательно один за другим. После приёма очередного пакета *контроллер* выполняет содержащиеся в нём команды и высылает пакет с ответом.



Обмен осуществляется посылками текстовых сообщений в формате JSON. Описание формата можно узнать, перейдя по ссылке. Посылки шифруются блочным потоковым шифром Blowfish.

Формат посылки.

Общий формат посылки приведен на рисунке.

Соль: случайное содержимое для инициализации кодера, 4 байта.

МН: маркер начала, 1 байт, 0x12.

JSON: текст команды в формате JSON.

МК: маркер конца, 1 байт, 0x14.

Выравнивающий блок: массив байт размером от 8 до 15, заполненный значением 0x16.

Используемые объекты и типы данных.

Объект “Command”.

Корневой объект, каждая посылка должна содержать этот объект. В зависимости от команды может содержать дополнительные поля разных типов. Ниже приведены обязательные поля для данного объекта.

Поля объекта:

- **fid** – идентификатор посылки, тип “String”.
- **cmnd** – идентификатор команды, тип “String”.
- **result** – идентификатор результата, тип “String”. Данное поле не является обязательным для запроса, содержится в ответе.

Объект “Bookmark”.

Объект представляет собой указатель на запись (пользователь, запись в протоколе и т.д.) в базе данных.

Поля объекта:

- **position** – идентификатор позиции, тип “String”. Может принимать следующие значения:
  - **begin** – указатель на первый объект в базе.
  - **end** – указатель на последний объект в базе.
  - **set** – указатель на конкретный объект в базе.
- **offset** – идентификатор смещения позиции объекта относительно первого объекта в базе, тип “Integer”. Используется только совместно с “position”:“set”.

Объект “BoxPermission”.

Объект представляет собой описание пенала/ячейки, к которой предоставлен доступ.

Поля объекта:

- **section** – идентификатор секции хранения, тип “Integer”.
- **box** – идентификатор пенала/ячейки, тип “Integer”.
- **schedule_mask** – идентификатор маски расписаний доступа, тип “Array”, элементы массива имеют тип “Integer”.
-

Объект “FingerPrint”.

Объект представляет собой описание отпечатка пальца пользователя.

Поля объекта:

- **finger** – идентификатор отпечатка, тип “Integer”.
- **length** – длина двоичных данных образа отпечатка, тип “Integer”.
- **data** – данные в виде кодированной строки “Base64”, тип “String.

Объект “BoxState”.

Объект представляет собой описание состояния пенала/ячейки.

Поля объекта:

- **box** – идентификатор пенала/ячейки, тип “Integer”.
- **state** – идентификатор состояния пенала/ячейки, тип “String”.

Объект “Time”.

Объект представляет собой информацию о времени.
Поля объекта:

- **year** – идентификатор года, тип “Integer”.
- **month** – идентификатор месяца, тип “Integer”.
- **day –** идентификатор дня месяца, тип “Integer”.
- **hour** – идентификатор часа, тип “Integer”.
- **minute** – идентификатор минуты, тип “Integer”.
- **second** – идентификатор секунды, тип “Integer”.

Объект “Interval”.
Объект представляет собой описание интервала времени.
Поля объекта:

- **start** – идентификатор начала интервала, тип “Time”.
- **end** – идентификатор конца интервала, тип “Time”.

Объект “Day”.
Объект представляет собой описание дня недели, как элемента расписания доступа.
Поля объекта:

- **day** – идентификатор дня недели, тип “String”. Может принимать следующие значения:
  - **Mon**
  - **Tue**
  - **Wed**
  - **Thu**
  - **Fri**
  - **Sat**
  - **Sun**
  - **Holiday**
  - **Spec1**
  - **Spec2**
- **interval** – идентификатор интервала времени доступа, тип “Interval”.

Объект “IdentificationType”.
Объект представляет собой описание метода авторизации, используемого секцией управления.
Поля объекта:

- **type** – идентификатор метода авторизации, тип “String”.

Объект “SectionDescription”.
Объект представляет собой описание секции хранения.
Поля объекта:

- **rows** – идентификатор числа строк в сетке ячеек секции, тип “Integer”.
- **columns** – идентификатор числа столбцов в сетке ячеек секции, тип “Integer”.
- **section** – идентификатор номера секции хранения, тип “Integer”.
- **type** – идентификатор типа секции хранения, тип “String”.
- **connected** – идентификатор состояния подключения секции хранения, тип “Boolean”.

Объект “ BoxConfig ”.

Объект представляет собой описание конфигурации ячейки. Поля объекта:

- **section** – идентификатор номера секции хранения, тип “Integer”.
- **box** – идентификатор номера пенала/ячейки, тип “ Integer ”.
- **name** – идентификатор типа секции хранения, тип “String”.
- **overtime** – идентификатор времени предупреждения, тип “Time”, используются только поля «**hour»** и **«minute»**.

Объект может не содержать поле «**name»** или **«overtime»**, в этом случае отсутствующий параметр не меняется. Для снятия параметров выдачи предупреждения необходимо параметру **overtime** присвоить значение **null** (“overtime”: null).

Объект “ BoxList ”.

Объект представляет собой идентификатор ячейки. Поля объекта:

- **section** – идентификатор номера секции хранения, тип “Integer”.
- **box** – идентификатор номера пенала/ячейки, тип “ Integer ”.

Объект “Data”.

Объект представляет собой блок данных (массив байт) различного назначения.

Поля объекта:

- **length** – длина двоичных данных, тип “Integer”.
- **data** – данные в виде кодированной строки “Base64”, тип “String.

Команды и параметры.

GetDevID

Запрос идентификатора устройства.

Не содержит дополнительных параметров.

Пример запроса в формате JSON:

{"fid":"0048ce_2B2E18", "cmnd":"getdevid"}

Ответ содержит дополнительное поле:

- **dev_id** – идентификатор устройства, тип “Строка”.

Пример ответа в формате JSON:

{"fid": "0048ce_2B2E18", "cmnd": "getdevid", "dev_id": "EVS_KSS_SKS ", "result": "OK"}

GetDevVersion

Запрос версии программного обеспечения устройства.

Не содержит дополнительных параметров.

Пример запроса в формате JSON:

{"fid":"0048ce_2B2E19", "cmnd":"getdevversion"}

Ответ содержит дополнительные поля:

- **version** – идентификатор версии, тип “Строка”.
- **build_date** – идентификатор даты выпуска версии, тип “Строка”.

Пример ответа в формате JSON:

{"fid": "0048ce_2B2E19", "cmnd": "getdevversion", "version": "7.11.8", "result": "OK", "build_date": "20131220"}

Synchronize

Команда синхронизации. При обмене через Ethernet использование данной команды не обязательно, команда присутствует для совместимости.

Не содержит дополнительных параметров.

Пример запроса в формате JSON:

{"fid":"0048ce_2B2E18", "cmnd":"synchronize"}

Ответ не содержит дополнительных полей.

Пример ответа в формате JSON:

{"fid": "0048ce_2B2E18", "cmnd": "synchronize", "result": "OK"}

GetConfig

Команда запроса текущей конфигурации секции управления.

Не содержит дополнительных параметров.

Пример запроса в формате JSON:

{"fid":"0048ce_2B2E1A", "cmnd":"getconfig"}

Ответ содержит дополнительные поля:

- **serial_number** – идентификатор серийного номера секции управления, тип “String”.
- **identification** – идентификатор метода авторизации, тип “Array”, тип элементов “IdentificationType”.

- **type** – идентификатор типа секции управления, тип “String”.
- **version** – идентификатор версии, тип “String”.
- **sections** – идентификатор числа секций хранения, тип “Integer”.
- **sections_list** – идентификатор подключенных секций, тип “Array”, тип элементов “SectionDescription”.

Пример ответа в формате JSON:

{"fid": "0048ce_2B2E1A", "serial_number": "604FCC05", "identification": [{"type": "card"}], "cmnd": "getconfig", "type": "sk24", "version": "7.11.8", "sections": 3, "sections_list": [{"columns": 3, "rows": 8, "section": 1, "type": "sk24", "connected": true}, {"columns": 3, "rows": 6, "section": 2, "type": "sd18", "connected": true}, {"columns": 4, "rows": 8, "section": 3, "type": "sk32", "connected": false}], "result": "OK"}

GetTime

Команда запроса текущего времени устройства.

Не содержит дополнительных параметров.

Пример запроса в формате JSON:

{"fid":"0048ce_2B2E1B", "cmnd":"gettime"}

Ответ содержит дополнительное поле:

- **time** – идентификатор текущего времени, тип “Time”.

Пример ответа в формате JSON:

{"fid": "0048ce_2B2E1B", "cmnd": "gettime", "time": {"second": 2, "year": 2014, "hour": 13, "month": 5, "minute": 29, "day": 23}, "result": "OK"}

SetTime

Команда установки текущего времени устройства.

Содержит дополнительный параметр:

- **time** – идентификатор текущего времени, тип “Time”.

Пример запроса в формате JSON:

{"fid":"0048ce_2B2E28", "cmnd":"settime", "time": {"year": 2014, "month": 05, "day": 23, "hour": 13, "minute": 35, "second": 14}

Ответ не содержит дополнительных полей.

Пример ответа в формате JSON:

{"fid": "0048ce_2B2E18", "cmnd": "settime", "result": "OK"}

GetEventLog

Команда запроса записи из базы данных протокола.

Может содержать дополнительный параметр:

- **bookmark** – идентификатор указателя на запись, тип “Bookmark”.

Если команда не содержит данного параметра, то возвращается следующая относительно текущего указателя запись, после чего указатель сдвигается. Если параметр указывает за границы диапазона записей, то возвращается ближайшая к указателю запись.

Примеры запросов в формате JSON:

{"fid":"0048ce_2B2E2C", "cmnd":"geteventlog", "bookmark": {"position": "begin"}

{"fid":"0048ce_2B2E2D", "cmnd":"geteventlog", "bookmark": {"position": "set", "offset": 1234}

{"fid":"0048ce_2B2E2E", "cmnd":"geteventlog"}

Ответ содержит дополнительные поля:

- **time** – идентификатор времени события, тип “Time”.
- **bookmark** – идентификатор указателя на запись, тип “Bookmark”.
- **event_no** – идентификатор события, тип “Integer”.

В зависимости от значения в поле **event_no** ответ может содержать одно или несколько из следующих полей:

- **section** – идентификатор секции хранения, с которой связано событие, тип “Integer”.
- **box** – идентификатор пенала/ячейки, с которым связано событие, тип “Integer”.
- **user_id** – идентификатор пользователя, с которым связано событие, тип “String”.

Пример ответа в формате JSON:

{"fid": "0048ce_2B2E41", "cmnd": "geteventlog", "time": {"second": 20, "year": 2014, "hour": 16, "month": 5, "minute": 5, "day": 20}, "bookmark": {"offset": 2236}, "event_no": 8, "section": 2, "box": 1, "user_id": "0000009BD6C3", "result": "OK"}

Идентификаторы событий:

0	Неизвестное событие
1	Пропала сеть
2	Появилась сеть
3	Аккумулятор разрядился
4	Датчик стены
5	Админ. выкл. сирену
6	Ячейка закрыта
7	Яч. открыта без разрешения
8	Ячейка открыта
9	Начало идентификации
10	Действие идентификации истекло
11	Настало время предупр.
12	Вставили чужой
13	Вставили свой без разреш.
14	Вынули чужой
15	Вынули свой, ошиб. сданный
16	Вставили в блокир. ячейку
17	Включили питание
18	Набран номер на клавиатуре
19	Введен неверный pin-code
20	Предоставлено разрешение
21	Карта блокирована
22	Нарушено расп. для яч.
23	Изм. ур. доступа для карты
24	Изм. список ячеек карты
25	Изм. маска расп. ячейки
26	Изм. время предупр. яч.
27	Изм. расписание
28	Таймаут ячейки
29	Прервалась связь с секцией
30	Установлена связь с секцией
31	Подтверждение доступа
32	Нет доступа к ячейке

GetUser

Команда запроса данных о пользователе.

Может содержать **один** из дополнительных параметров:

- **bookmark** – идентификатор указателя на запись, тип “Bookmark”.
- **user_id** – идентификатор пользователя, тип “String”.

Если команда не содержит ни одного из параметров, то возвращается следующая относительно текущего указателя запись, после чего указатель сдвигается.

Примеры запросов в формате JSON:

{"fid":"0048ce_2B2E43", "cmnd":"getuser", "bookmark": {"position": "begin"}}
{"fid":"0048ce_2B2E44", "cmnd":"getuser"}
{"fid":"0048ce_2B2E45", "cmnd":"getuser", "user_id": "0000000000004457"}

Ответ содержит дополнительные поля:

- **user_id** – идентификатор пользователя, тип “String”.
- **level** – идентификатор уровня доступа пользователя, тип “String”. Может принимать следующие значения:
  - **no_user** – пользователь отсутствует в базе
  - **user** – пользователь
  - **superuser** – дежурный
  - **admin** – администратор
- **boxes_list** – идентификатор списка назначенных пеналов/ячеек, тип “Array”, тип элементов “BoxPermission”.

Примеры ответов в формате JSON:

{"fid": "0048ce_2B2E44", "cmnd": "getuser", "user_id": "0000001EE9DC", "boxes_list": [{"section": 2, "box": 2, "schedule_mask": [1]}], "level": "user", "result": "OK"}

{"fid": "0048ce_2B2E45", "cmnd": "getuser", "user_id": "000000004457", "level": "no_user", "result": "OK"}

AddUser

Команда добавления пользователя в устройство.

Содержит дополнительные параметры:

- **user_id** – назначенный идентификатор пользователя, тип “String”;
  поле должно содержать код карты или личный номер в формате:
  Код карты – строка 16 символов, в которой в шестнадцатиричном формате указан код карт, выравнивание должно быть слева (для кода карты **2B2E45 –** строка **00000000002B2E45**);
  Личный номер – строка 16 символов, в которой первые 8 символов длина кода (для личного номера **12345** – строка **0000000500012345**).
- **level** – уровень доступа пользователя, тип “String”.
- **boxes_list** – идентификатор списка назначенных пеналов/ячеек, тип “Array”, тип элементов “BoxPermission”.

Пример запроса в формате JSON:

{"fid": "0048ce_2B2E46", "cmnd": "adduser", "user_id": "0000000000004457", "level": "superuser", "boxes_list": [{"section": 1, "box": 13, "schedule_mask": [1,3,9]}, {"section": 5, "box": 2}]}

Ответ не содержит дополнительных полей.

Пример ответа в формате JSON:

{"fid": "0048ce_2B2E46", "cmnd": "adduser", "result": "OK"}

DelUser

Команда удаления пользователя из устройства.

Содержит дополнительный параметр:

- **user_id** – идентификатор пользователя, тип “String”.

Пример запроса в формате JSON:

{"fid":"0048ce_2B2E47", "cmnd":"deluser", "user_id": "0000000000004457"}

Ответ не содержит дополнительных полей.

Пример ответа в формате JSON:

{"fid": "0048ce_2B2E47", "cmnd": "deluser", "result": "OK"}

DelAllUsers

Команда удаления всех пользователей из устройства.

Не содержит дополнительных параметров.

Пример запроса в формате JSON:

{"fid":"0048ce_2B2E48", "cmnd":"delallusers"}

Ответ не содержит дополнительных полей.

Пример ответа в формате JSON:

{"fid": "0048ce_2B2E48", "cmnd": "delallusers", "result": "OK"}

GetBoxesState

Команда запроса состояния пеналов/ячеек в секции хранения.

Содержит дополнительный параметр:

- **section** – идентификатор секции хранения, тип “Integer”.

Пример запроса в формате JSON:

{"fid":"005015_2C65DB", "cmnd":"getboxesstate", "section":2}

Ответ содержит дополнительные поля:

- **section** – идентификатор секции хранения, тип “Integer”.
- **type** – идентификатор типа секции хранения, тип “String”.
- **state_list** – идентификатор состояния ячеек, тип “Array”, тип элементов “BoxState”.

Пример ответа в формате JSON:

{"fid": "005015_2C65DB", "cmnd": "getboxesstate", "section": 2, "type": "sd18", "state_list": [{"box": 1, "state": "closed"}, {"box": 2, "state": "closed"}, {"box": 3, "state": "closed"}, {"box": 4, "state": "closed"}, {"box": 5, "state": "closed"}, {"box": 6, "state": "closed"}, {"box": 7, "state": "closed"}, {"box": 8, "state": "closed"}, {"box": 9, "state": "closed"}, {"box": 10, "state": "closed"}, {"box": 11, "state": "closed"}, {"box": 12, "state": "closed"}, {"box": 13, "state": "closed"}, {"box": 14, "state": "closed"}, {"box": 15, "state": "closed"}, {"box": 16, "state": "closed"}, {"box": 17, "state": "closed"}, {"box": 18, "state": "closed"}], "result": "OK"}

Возможные состояния ячеек секций хранения CX24/CX32:

- **empty_box** - пенал не сдан
- **inserted_no_authorization** - пенал сдан без авторизации
- **inserted** - пенал сдан
- **alien_inserted** - сдан чужой пенал
- **alarm** - пенал взят без авторизации
- **overtimed** - пенал не сдан вовремя

Возможные состояния ячеек секций хранения СХП18:

- **opened** - ячейка открыта
- **closed_no_authorization** - ячейка закрыта без авторизации
- **closed** - ячейка закрыта
- **alarm** - взлом ячейки

GetSchedule

Команда запроса параметров расписания.

Содержит дополнительный параметр:

- **schedno** – идентификатор расписания, тип “Integer”.

Пример запроса в формате JSON:

{"fid":"005015_2C65DC", "cmnd":"getschedule", "schedno":1}

Ответ содержит дополнительные поля:

- **schedno** – идентификатор расписания, тип “Integer”.
- **days_list** – идентификатор списка дней, использованных в расписании, тип “Array”, тип элементов “Day”.

Пример ответа в формате JSON:

{"fid": "005015_2C65DC", "cmnd": "getschedule", "schedno": 1, "days_list": [{"day": "Mon", "interval": {"start": {"hour": 0, "minute": 0}, "end": {"hour": 23, "minute": 59}}}, {"day": "Tue", "interval": {"start": {"hour": 0, "minute": 0}, "end": {"hour": 23, "minute": 59}}}, {"day": "Wed", "interval": {"start": {"hour": 0, "minute": 0}, "end": {"hour": 23, "minute": 59}}}, {"day": "Thu", "interval": {"start": {"hour": 0, "minute": 0}, "end": {"hour": 23, "minute": 59}}}, {"day": "Fri", "interval": {"start": {"hour": 0, "minute": 0}, "end": {"hour": 23, "minute": 59}}}, {"day": "Sat", "interval": {"start": {"hour": 0, "minute": 0}, "end": {"hour": 23, "minute": 59}}}, {"day": "Sun", "interval": {"start": {"hour": 0, "minute": 0}, "end": {"hour": 23, "minute": 59}}}], "result": "OK"}

SetSchedule

Команда установки параметров расписания в устройстве.

Содержит дополнительные параметры:

- **schedno** – идентификатор расписания, тип “Integer”.
- **days_list** – идентификатор списка дней, использованных в расписании, тип “Array”, тип элементов “Day”.

Пример запроса в формате JSON:

{"fid":"005015_2C65DD", "cmnd":"setschedule", "schedno":2, "days_list": [{"day": "Mon", "interval": {"start": {"minute": 30, "hour": 12}, "end": {"minute": 0, "hour": 15}}}, {"day": "Sun", "interval": {"start": {"minute": 5, "hour": 1}, "end": {"minute": 58, "hour": 16}}}]}

Ответ содержит дополнительное поле:

- **schedno** – идентификатор расписания, тип “Integer”.

Пример ответа в формате JSON:

{"fid": "005015_2C65DD", "cmnd": "setschedule", "result": "OK", "schedno": 2}

ShowMsg

Команда вывода на дисплей устройства произвольного сообщения.

Содержит дополнительный параметр:

- **msg** – идентификатор отображаемого сообщения, тип “Array”, тип элементов “String”.

Пример запроса в формате JSON:
{"fid":"005015_2C65DE", "cmnd":"showmsg", "msg":["Line 1","Line 2","Line 3"]}

Ответ не содержит дополнительных полей.
Пример ответа в формате JSON:
{"fid": "005015_2C65DE", "cmnd": "showmsg", "result": "OK"}

GetAlarmTime
Команда запроса параметров выдачи предупреждения о несданном пенале.
Содержит дополнительные параметры:

- **section** – идентификатор секции хранения, тип “Integer”.
- **box** – идентификатор пенала, тип “Integer”.

Пример запроса в формате JSON:
{"fid":"005015_2C65E1", "cmnd":"getalarmtime", "section": 1, "box": 15}

Ответ содержит дополнительные поля:

- **section** – идентификатор секции хранения, тип “Integer”.
- **box** – идентификатор пенала, тип “Integer”.
- **time** – идентификатор времени уведомления, тип “Time”.

Пример ответа в формате JSON:
{"fid": "005015_2C65E1", "cmnd": "getalarmtime", "section": 1, "box": 15, "time": {"hour": 18, "minute": 50}, "result": "OK"}

SetAlarmTime
Команда установки параметров выдачи предупреждения о несданном пенале.
Содержит дополнительные параметры:

- **section** – идентификатор секции хранения, тип “Integer”.
- **box** – идентификатор пенала, тип “Integer”.
- **time** – идентификатор времени уведомления, тип “Time”.

Для снятия параметров выдачи предупреждения необходимо параметру **time** присвоить значение **null** (“time”: null).
Пример запроса в формате JSON:
{"fid":"005015_2C65E2", "cmnd":"setalarmtime", "section": 1, "box": 15, "time": {"hour": 18, "minute": 50}}

Ответ не содержит дополнительных полей.
Пример ответа в формате JSON:
{"fid": "005015_2C65E2", "cmnd": "setalarmtime", "result": "OK"}

OpenCell
Команда открытия пенала/ячейки.
Содержит дополнительные параметры:

- **section** – идентификатор секции хранения, тип “Integer”.
- **box** – идентификатор пенала, тип “Integer”.

Пример запроса в формате JSON:
{"fid":"005015_2C65E7", "cmnd":"opencell", "section": 1, "box": 15}

Ответ не содержит дополнительных полей.
Пример ответа в формате JSON:
{"fid":"005015_2C65E7", "cmnd":"opencell", “result”: “OK”}

CloseCell
Команда закрытия всех ячеек.
Может содержать дополнительные параметры:

- **clearalarm** – если значение равно нулю, то дополнительно выполняется команда сброса «Тревоги», тип “Integer”.

Пример запроса в формате JSON:
{"fid":"005015_2C65E8", "cmnd":"closecell"}

Ответ не содержит дополнительных полей.
Пример ответа в формате JSON:
{"fid": "005015_2C65E8", "cmnd": "closecell", "result": "OK"}

GetEventLog2

Команда запроса записи из базы данных протокола. Поддерживается устройствами с прошивкой версии 8.0.0 и выше. Запрос не отличается от команды **GetEventLog**.

В зависимости от идентификатора события в ответе дополнительно может присутствовать:

- **card_code –** код карты, связанной с событием, тип “String”;
- **number –** личный номер, связанной с событием, тип “String”.

Для события (46) добавиться поле **card_code.** Для события (9) – в зависимости от способа идентификации либо **card_code** , либо **– number.**

Идентификаторы событий:

0	Неизвестное событие
1	Пропала сеть
2	Появилась сеть
3	Аккумулятор разрядился
4	Датчик стены
5	Админ. выкл. сирену
6	Ячейка закрыта
7	Яч. открыта без разрешения
8	Ячейка открыта
9	Начало идентификации
10	Действие идентификации истекло
11	Настало время предупр.
12	Вставили чужой
13	Вставили свой без разреш.
14	Вынули чужой
15	Вынули свой, ошиб. сданный
16	Вставили в блокир. ячейку
17	Включили питание
18	Набран номер на клавиатуре
19	Введен неверный pin-code
20	Предоставлено разрешение
21	Карта блокирована
22	Нарушено расп. для яч.
23	Изм. ур. доступа для карты
24	Изм. список ячеек карты
25	Изм. маска расп. ячейки
26	Изм. время предупр. яч.
27	Изм. расписание
28	Таймаут ячейки
29	Прервалась связь с секцией
30	Установлена связь с секцией
31	Подтверждение доступа

32	Нет доступа к ячейке
33	Изменение параметров пользователя
34	Изменение личного номера пользователя
35	Изменение карты пользователя
36	Изменение ПИН-кода пользователя
37	Установлено соединение через порт RS-485
38	Соединение на порту RS-485 потеряно
39	Установлено соединение через порт Ethernet
40	Соединение на порту Ethernet потеряно
41	В цепи аккумулятора неисправность
42	Открыта дверца ячейки
43	Закрыта дверца ячейки
44	Ячейка открыта без разрешения
45	Ячейка закрыта без разрешения
46	Предъявлена незарегистрированная карта
47	Введен незарегистрированный личный номер
48	Незарегистрированный отпечаток пальца

GetSchedule2

Команда запроса параметров расписания. Поддерживается устройствами с прошивкой версии 8.0.0 и выше. Запрос не отличается от команды **GetSchedule**.

В ответе дополнительно содержится:

- **schedname** – имя расписания, тип “String”.

SetSchedule2

Команда установки параметров расписания в устройстве. Поддерживается устройствами с прошивкой версии 8.0.0 и выше.

Дополнительно к **SetSchedule** содержит следующие поля:

- **schedname** – имя расписания, тип “String”.

Ответ не отличается от ответа на команду **SetSchedule**.

Имя расписания и ФИО в устройстве может содержать только заглавные буквы русского и английского алфавитов, цифры, пробелы, тире и апострофы. При передаче буквы русского алфавита должны быть закодированы в соответствии с таблицей ниже. Буквы Е и Ё считаются одинаковыми. Буквы английского алфавита, цифры и допустимые символы не кодируются.

А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	!	@	#	$	%	^

При приеме требуется обратное преобразование.

GetUser2

Команда запроса данных о пользователе. Поддерживается устройствами с прошивкой версии 8.0.0 и выше.

Содержит дополнительный параметр:

- **user_id** – идентификатор пользователя, тип “String”. Данное поле является обязательным. Для запроса первого пользователя значение данного поля должно быть равно “0”. В ответе в данном поле вернется значение идентификатора первого пользователя в устройстве. Для запроса следующего пользователя необходимо инкрементировать полученное значение и вставить его в следующую команду **GetUser2**.

Пример запроса в формате JSON:

{"fid":"14","cmnd":"getuser2","user_id":"0"}

Ответ содержит дополнительные поля:

- **user_id** – идентификатор пользователя, тип “String”.
- **level** – идентификатор уровня доступа пользователя, тип “String”. Может принимать следующие значения:
  - **no_user** – пользователь отсутствует в базе
  - **user** – пользователь
  - **superuser** – дежурный
  - **admin** – администратор
- **boxes_list** – идентификатор списка назначенных пеналов/ячеек, тип “Array”, тип элементов “BoxPermission”.

Кроме того, ответ может содержать следующие поля:

- **username** – имя пользователя, тип “String”.
- **usersoname** – отчество пользователя, тип “String”.
- **userfamily** – фамилия пользователя, тип “String”.
- **card** – код карты, назначенный пользователю, тип “String”.
- **number** – личный номер пользователя, тип “String”.
- **userbirthday** – год рождения пользователя, тип “String”.
- **fingers** – идентификатор списка назначенных отпечатков пальцев, тип “Array”, тип элементов “FingerPrint”.

Поля **username**, **usersoname**, **userfamily** закодированы по тому же правилу, что и поле **schedname** команд **GetSchedule2** и **SetSchedule2**.

Примеры ответов в формате JSON:

{"fid": "14", "cmnd": "getuser2","userfamily": "kalinxfc", "level": "admin", "boxes_list": [{"section": 0, "box": 4, "schedule_mask": [0]}], "user_id": "1", "card": "1ee9dc", "number": "123", "result": "OK"}

{"fid": "38", "cmnd": "getuser2", "user_id": null, "level": "no_user", "result": "OK"}

AddUser2

Команда добавления пользователя в устройство и редактирования его данных. Поддерживается устройствами с прошивкой версии 8.0.0 и выше.

Содержит дополнительные параметры:

- **user_id** – идентификатор пользователя, тип “String”. Для добавления нового пользователя значение данного поля должно быть “0”. Если данное поле содержит другое значение – устройство выполнит попытку отредактировать данные пользователя с указанным идентификатором, новый пользователя создан не будет, даже если пользователя с таким идентификатором в устройстве нет.
- **level** – уровень доступа пользователя, тип “String”.
- **boxes_list** – идентификатор списка назначенных пеналов/ячеек, тип “Array”, тип элементов “BoxPermission”.

Кроме того, запрос может содержать следующие поля:

- **username** – имя пользователя, тип “String”.
- **usersoname** – отчество пользователя, тип “String”.
- **userfamily** – фамилия пользователя, тип “String”.
- **card** – код карты, назначенный пользователю, тип “String”.
- **number** – личный номер пользователя, тип “String”.

- **userbirthday** – год рождения пользователя, тип “String”.
- **fingers** – идентификатор списка назначенных отпечатков пальцев, тип “Array”, тип элементов “FingerPrint”.
- **userdata** – пользовательские данные, тип “String”. Произвольные данные, устройство не обрабатывает данное поле, возвращает его в ответе в том же виде, в каком получило.

Поля **username**, **usersoname**, **userfamily** закодированы по тому же правилу, что и поле **schedname** команд **GetSchedule2** и **SetSchedule2**.

Пример запроса в формате JSON:

{"fid": "14", "cmnd": "adduser2","userfamily": "kalinxfc", "level": "admin", "boxes_list": [{"section": 0, "box": 4, "schedule_mask": [0]}], "user_id": "0", "card": "1ee9dc", "number": "123","userdata": "c45ddafedb5"}

Ответ содержит дополнительные поля:

- **user_id –** идентификатор пользователя, тип “String”. Если в результате команды создан новый пользователь – будет указан идентификатор вновь созданного пользователя.
- **userdata** – пользовательские данные, тип “String”.
- **card** – код карты, если данный код карты уже зарегистрирован, тип “String”.
- **number** – личный номер пользователя, если данный личный номер уже зарегистрирован, тип “String”.
- **existing_user_id** – идентификатор пользователя, тип “String”. Если в результате команды обнаружен код карты или личный номер, который уже зарегистрирован на другого пользователя. То это поле этого содержит идентификатор пользователя.

Пример ответа в формате JSON:

{"fid": "14", "cmnd": "adduser2","user_id": "1","","userdata": " c45ddafedb5","result": "OK"}

DelUser2

Команда удаления пользователя из устройства. Поддерживается устройствами с прошивкой версии 8.0.0 и выше.

Если необходимо удалить всех пользователей в базе **user_id** должен быть равен 0 и пакет должен содержать поле **delpass**.

Содержит дополнительный параметр:

- **user_id** – идентификатор пользователя, тип “String”.

Кроме того, запрос может содержать следующие поля:

- **userdata** – пользовательские данные, тип “Integer”.
- **delpass** – пароль удаления всех пользователей, тип “String”. Пароль должен быть равен 23043015.0

Пример запроса в формате JSON:

{"fid":"14", "cmnd":"deluser2", "user_id": "1", "userdata": " c45ddafedb5"}

Ответ может содержать дополнительные поля:

- **userdata** – пользовательские данные, тип “String”.

Пример ответа в формате JSON:

{"fid": "14", "cmnd": "deluser2", ", "userdata": " c45ddafedb5","result": "OK"}

SetConfigBox

Команда установки конфигурации ячейки устройства. Поддерживается устройствами с прошивкой версии 8.2.42 и выше.

С помощью этой команды можно задать имя и время предупреждения для ячейки.

Содержит дополнительные параметры:

- **boxes** – список конфигураций ячеек, тип “Array”, тип элементов “BoxConfig”.

В дополнительных параметрах:

Элемент “BoxConfig” может не содержать поле «**name»** или **«overtime»**, в этом случае отсутствующий параметр не меняется. Для снятия параметров выдачи предупреждения необходимо параметру **overtime** присвоить значение **null** (“overtime”: null).

Пример запроса в формате JSON:

{"fid":"14","cmnd":“setconfigbox”,“boxes”:[{“section”:1,”box”:1,”name”:”k102”, ”overtime”:{“hour”:17,”minute”:0} }, …] }

Пример ответа в формате JSON:

{"fid": "14", "cmnd": "setconfigbox", ", "result": "OK"}

GetConfigBox

Команда установки конфигурации ячейки устройства. Поддерживается устройствами с прошивкой версии 8.2.42 и выше.

С помощью этой команды можно запросить заданные имя и время предупреждения для ячейки.

Запрос не содержит дополнительных параметров. Ответ будет содержать только те ячейки, для которых задан хоть один параметр.

Ответ содержит параметры:

- **boxes** – список конфигураций ячеек, тип “Array”, тип элементов “BoxConfig”;
- **result** – результат выполнения, тип “String”.

Пример запроса в формате JSON:

{"fid":"14","cmnd":"getconfigbox"}

Пример ответа в формате JSON:

{"fid": "14", "cmnd": “setconfigbox”,“boxes”:[{“section”:1,”box”:1,”name”:”k102”, ”overtime”:{“hour”:17,”minute”:0} }, …], "result": "OK"}

ShowSelectCell

Команда для управления прибором в жестком режиме (полного управления).

По этой команде «СК-12» предложит доступ пользователю к ячейке из списка. Если в списке одна ячейка, то она автоматически откроется. Команда выполняется только при типе внешнего управления «Полное».

Поддерживается устройствами с прошивкой версии 8.2.51 и выше.

Содержит дополнительные параметры:

- **user_name** – назначенный идентификатор пользователя, тип “String”.

  Поля может не быть.

  Поле должно быть закодировано по тому же правилу, что и поле **schedname** команд **GetSchedule2** и **SetSchedule2**;
- **options** – параметры выполнения команды, тип “Integer”:
  - *0 бит* – зарезервирован;

- *1 бит* – установлен, если пользователь имеет права дежурного;
- *2 бит* – установлен, если пользователь имеет права администратора;
- *3 и 4 биты* – 01 (3-й установлен), то при выборе нужной ячейки произойдет немедленный выход в режим ожидания новой карты или номера; 10, то - тоже самое, но у пользователя не будет возможности отменить немедленный выход; 11 – зарезервировано.
- *5 бит* – установлен, при этом меню с выбором ячейки не появляется и «СК-12» издает звук отказа доступа;
- *6 бит* – установлен, при выборе пользователем ячейки, «СК-12» формирует событие «Выбрана ячейка», но не дает доступа к ячейке;
- *7 бит* – установлен, будет немедленный выход в режим ожидания новой карты или номера;

- **boxes_list** – идентификатор списка назначенных пеналов/ячеек, тип “Array”, тип элементов “BoxList”.

Пример запроса в формате JSON:

{"fid": "0048ce_2B2E46", "cmnd": "showselectcell", "user_name": "Name", "options": 0, "boxes_list": [{"section": 1, "box": 13}, {"section": 5, "box": 2}]}

Ответ не содержит дополнительных полей.

Пример ответа в формате JSON:

{"fid": "0048ce_2B2E46", "cmnd": " showselectcell ", "result": "OK"}

AddUser3

Команда добавления пользователя в устройство и редактирования его данных. Поддерживается устройствами с прошивкой версии 8.3.0 и выше.

Содержит дополнительные параметры:

- **user_id** – идентификатор пользователя, тип “String”. Для добавления нового пользователя значение данного поля должно быть “0”. Если данное поле содержит другое значение – устройство выполнит попытку отредактировать данные пользователя с указанным идентификатором, новый пользователя создан не будет, даже если пользователя с таким идентификатором в устройстве нет.
- **level** – уровень доступа пользователя, тип “String”.
- **boxes_list** – идентификатор списка назначенных пеналов/ячеек, тип “Array”, тип элементов “BoxPermission”.

Кроме того, запрос может содержать следующие поля:

- **username** – имя пользователя, тип “String”.
- **usersoname** – отчество пользователя, тип “String”.
- **userfamily** – фамилия пользователя, тип “String”.
- **card** – код карты, назначенный пользователю, тип “String”.
- **number** – личный номер пользователя, тип “String”.
- **userbirthday** – год рождения пользователя, тип “String”.
- **fingers** – идентификатор списка назначенных отпечатков пальцев, тип “Array”, тип элементов “FingerPrint”.
- **photo** – фотография пользователя, тип элементов “Data”.
- **userdata** – пользовательские данные, тип “String”. Произвольные данные, устройство не обрабатывает данное поле, возвращает его в ответе в том же виде, в каком получило.

Поля **username**, **usersoname**, **userfamily** закодированы по тому же правилу, что и поле **schedname** команд **GetSchedule2** и **SetSchedule2**.

Пример запроса в формате JSON:

{"fid": "14", "cmnd": "adduser2","userfamily": "kalinxfc", "level": "admin", "boxes_list": [{"section": 0, "box": 4, "schedule_mask": [0]}], "user_id": "0", "card": "1ee9dc", "number": "123","userdata": "c45ddafedb5"}

Ответ содержит дополнительные поля:

- **user_id –** идентификатор пользователя, тип “String”. Если в результате команды создан новый пользователь – будет указан идентификатор вновь созданного пользователя.
- **userdata** – пользовательские данные, тип “String”.
- **card** – код карты, если данный код карты уже зарегистрирован, тип “String”.
- **number** – личный номер пользователя, если данный личный номер уже зарегистрирован, тип “String”.
- **existing_user_id** – идентификатор пользователя, тип “String”. Если в результате команды обнаружен код карты или личный номер, который уже зарегистрирован на другого пользователя. То это поле этого содержит идентификатор пользователя.

Пример ответа в формате JSON:

{"fid": "14", "cmnd": "adduser2","user_id": "1","","userdata": " c45ddafedb5","result": "OK"}

GetEventVideoLog

Команда запроса записи из базы данных протокола видеоролика авторизации пользователя. Поддерживается устройствами с прошивкой версии 8.3.0 и выше.

Содержит дополнительные параметры:

- **bookmark** – идентификатор указателя на запись в протоколе, тип “Bookmark”.

Пример ответа в формате JSON:

{"fid":"0048ce_2B2E20", "cmnd":"geteventvideolog", "bookmark": {"position": "set", "offset": 1234}

Ответ содержит дополнительные поля:

- **bookmark** – идентификатор указателя на запись, тип “Bookmark”;
- **video –** один или несколько кадров снятых камерой при авторизации пользователя, тип элементов “Data”. Если по данному «**bookmark»** нет видеокадров, то в ответе на команду длина блока данных будет равна нулю. Блок данных имеет следующую структуру:
  - **count –** количество кадров, тип Integer (4 байта, первый младший);
  - **length1 –** длина первого кадра, тип Integer (4 байта, первый младший);
  - **…**…..
  - **lengthX –** длина кадра с номером **X**, тип Integer (4 байта, первый младший);
  - **data –** массив байт содержащий кадры (сначала первый кард и далее остальные поочередно)

Пример ответа в формате JSON:

{"fid": "0048ce_2B2E20", "cmnd": " geteventvideolog ", "bookmark": {"position": "set", "offset": 1234}, "video":{"length":0}, "result": "OK"}