

Настройка декодирования для Mifare (подробно)

1. У тебя есть читалка (допустим z-2) и читалка турникета (matrix)
2. Главная задача это привести треки получаемые со считалок в один единый вид (в 10 формате)
3. Если
 - 3.1 Mifare
 - 3.1.1 Скачиваешь утилиту Putty
 - 3.1.2 Подключаешься через Putty к читалке
 - 3.1.3 Сканируешь карту и получаешь трек **Mifare[A1B2C3D4]** (какая то номерная последовательность, которая совершенно не интересна нам) <терминальные символы>
 - 3.1.4 Открываешь version.txt из каталогов TCPCardReader и decodercards (лежит в сборке в UTILS)
 - 3.1.5 ~~Начинаешь читать и разбираться какие параметры нужны~~
 - 3.1.6 Итак, сперва, чтобы настроить декодирование в TCPCardRead нам нужно отделить искомую последовательность от остальной информации. В данном случае необходимо вытащить UID из трека Mifare[**A1B2C3D4**] (какая то номерная последовательность, которая совершенно не интересна нам) <терминальные символы> **(выделено жирным, без скобок !)**.
 - 3.1.7 Открываем TcpCardReader.ini удаляем от туда секцию [TCPCARDREADER] (если есть (мы её не используем))
 - 3.1.8 Смотрим секции типа [READER1], [READER2], ... [READERn] - эти секции отвечаю за каждую свою подключённую читалку. Если нам нужна 1 читалка, то делаем секцию [READER1], если две, то записываем секцию [READER2] и т.д.
 - 3.1.9 Итак, настроим, например, только 1 считыватель. Смотрим секцию [READER1]

```
[ READER1]
COMPORT=9
COMPORTPARAMS=baud=4800 data=8 parity=N stop=1
;Параметр Prefix задаёт строку, которая передаётся клиентам перед первым прочтенным символом
Prefix=
;Параметр Postfix задаёт строку, которая передаётся клиентам после последнего прочтенного
символа
Postfix=

;Использовать decodercards.dll
```

```
useDecodeCards=0
StartTermChars=;
FinishTermChars=?#13; #10; #0;

; Файл логирования треков
TrackLogFile=
ConvertTrackLog=0
IncludeTypeStartTermChar=0
IncludeStartTermChar=0
NeedCheckAndReopenCom=1
```

3.1.10 Смотрим параметры из version.txt (например, через NotePad++)

```
{
COMPORT=9 - номер порта берется из диспетчера устройств. (диспетчер устройств ->
порты)
COMPORTPARAMS=baud=4800 data=8 parity=N stop=1 (параметры подключения)
практически никогда не меняется (берется из свойств порта. см. диспетчер устройств)
}
;Параметр Prefix задаёт строку, которая передаётся клиентам перед первым прочтенным
символом
Prefix=
```

(строка №7 из version.txt) (";Параметр Prefix задаёт строку, которая передаётся клиентам перед первым прочтенным символом") Это означает, что передаваемый трек клиентам будет начинаться с установленного префикса. Например, если предположим, что декодирование настроено, то пусть наш трек выглядит так 1200. Мы хотим передавать трек с префиксом ;, то устанавливаем в параметр Prefix=; При передаче трека клиентам (abonnementmanager, connecter, ...) передастся трек ;1200 . Если мы хотим передать трек с префиксом mrgreen, то ставим в параметрах Prefix=mrgreen. Соответственно при передаче трека клиентам, они получают mrgree1200

;Параметр Postfix задаёт строку, которая передаётся клиентам после последнего прочтенного символа
Postfix=

(строка №9 из version.txt) (";Параметр Postfix задаёт строку, которая передаётся клиентам после последнего прочтенного символа") По аналогии с суффиксом. Это означает, что передаваемый трек клиентам, оканчивается на установленный суффикс. Например, хотим передать суффикс ?, то Postfix=?. Клиенты получают трек 1200?. Если

хотим передать суффикс `?#`, то ставим `Postfix=?#`. Клиенты получают трек `1200@?#`. Таким образом, если установлены параметры `Prefix=;` и `Postfix=?`, то клиенты получают трек `;1200?`

;Использовать decodecards.dll
`useDecodeCards=0`

(строка №27 из `version.txt`) использовать= 1 - не использовать=0 `decodecards.dll` для извлечения номера. По умолчанию 0 - не использовать.

`StartTermChars=;`

(строка №28 из `version.txt`) "Стартовые символы. Используются, если `useDecodeCards=1`" Стартовые символы или управляющие. Это символ(ы), с которого идет наш необработанный трек, передаваемый в `decodercards`.

`FinishTermChars=?#13;#10;#0;`

(строка №29 из `version.txt`) Терминальные символы. Используются, если `useDecodeCards=1`. Это символ(ы), с которого идет наш необработанный трек, передаваемый в `decodercards`.

Т.е.
Допустим Трек **`Mifare[A1B2C3D4]`** (какая то номерная последовательность, которая совершенно не интересна нам) <терминальные символы>

(терминальный символ зависит от прошивки/настройки устройства, он передается вместе с данными, как правило это перевод на новую строку и возврат каретки и - №10 и №13 символы таблицы ASCII
<https://ru.wikipedia.org/wiki/ASCII> ; получаемая последовательность со считывателя в `TPCCardReader` или `Putty` выглядит наподобие `Mifare[A1B2C3D4]` (....) <LF><CR>).

Если сделаем **`StartTermChars=i`** и **`FinishTermChars=?#13;#10;#0;`** то будет следующее:

1. В треке находится начальный символ, указанный в **`StartTermChars`**, это **`i`**. В нашем случае **`M i fare[A1B2C3D4]`** (....) <LF><CR>

2. Затем ищется окончание (последний символ), указанные в **`FinishTermChars`** (можно указывать несколько подряд).

Т.е. смотрим на **`Mifare[A1B2C3D4]`** (....) <LF><CR>, есть ли в данном треке символ `"?"` - нету

есть ли символ `#13`; (синтаксис 13 элемента в таблице в `TCPCardReader` - это <CR> возврат

каретки), есть Mifare[A1B2C3D4] (....) <LF> <CR>

Ага, **последовательность найдена** и **результат будет** : fare[A1B2C3D4] (....) <LF> и он уже передается на обработку в decodercards.dll

;Файл логирования треков

TrackLogFile=

ConvertTrackLog=0

(строка №107 из version.txt)

Если TrackLogFile определен, то используется дополнительный параметр ConvertTrackLog.

Он определяет, каким образом логировать невизуальные символы. По умолчанию ConvertTrackLog=0 (не конвертировать).

Если ConvertTrackLog=1, то невизуальные символы конвертируются в их код.

IncludeTypeStartTermChar=0 - (строка №126 из version.txt)

1.1. IncludeTypeStartTermChar - тип обработки стартового символа

Имеет смысл, если useDecodeCards=1, IncludeStartTermChar=1 и UseOriginalTrack=0

Если IncludeTypeStartTermChar=0 (по умолчанию), то стартовый символ НЕ передается в decodercards, а подставляется перед истинным номером как дополнительный префикс

Если IncludeTypeStartTermChar=1, то стартовый символ передается в decodercards и участвует в извлечении истинного номера

Обычно используем по умолчанию

NeedCheckAndReopenCom=1 - (строка №135 из version.txt)

Если NeedCheckAndReopenCom=1 (по умолчанию), то постоянно контролируется состояние соединения. Если связь потеряна, то происходит попытка переподключиться.

NeedCheckAndReopenCom=0 можно устанавливать, если устройство подключается не к USB, а к COM-порту.

Оставляем 1.

3.1.11 Смотря пункт 3.1.10, составляем следующую конфигурацию в секции **[READER1]**

```
COMPORT=9
COMPORTPARAMS=baud=4800 data=8 parity=N stop=1
;Параметр Prefix задаёт строку, которая передаётся клиентам перед первым прочтенным символом
Prefix=;
;Параметр Postfix задаёт строку, которая передаётся клиентам после последнего прочтенного
символа
Postfix=?

;Использовать decodecards.dll
useDecodeCards=1
StartTermChars=i
FinishTermChars=?#13; #10; #0;
```

3.1.12 Итак, получили на входе в decodecards.dll трек на обработку **fare[A1B2C3D4] (....)**
<LF>

3.1.13 Разбираемся с декодированием. Открываем ini из сборки и видим

```
[ fr f]
fsUnit=fsdecodecard.upas

[ general]
usemask=1
TrackResultLog=

[ mask]
card = 05015005*
card2= 778=12345678=*
cardfile=file: //cards.txt

[ cards]
CardPrefix    = 05015005
;              05015005280507210
CardPrefix    = 750=
CardPrefix    = 778=999999999=
CardPrefix    = 778=123456789=
StaffCardPrefix      = 778=87121234=
StaffCardPrefix      = 778=201050001
RegularCardPrefix    = 778=444444444=
PDSCardPrefix    = 811q
```

```
[ cards. cardREPLACECARDNO]
```

```
ExcludedPrefix=
```

```
code=2
```

```
CardDecodeType = REPLACECARDNO
```

```
NewCardNo = 5678956
```

```
[ cards. ARRAYOFBYTES]
```

```
CardDecodeType = MASK
```

```
mask=*cccc*
```

```
bitmask=$7FFFFFFF
```

```
MaskType=ARRAYOFBYTES
```

```
code=2
```

```
[ cards. Card2]
```

```
ExcludedPrefix=778=12345678=
```

```
; CardDecodeType = ANGSTREMCARD
```

```
; CardDecodeType = ANGSTREMBRASLET
```

```
code=2
```

```
; CardDecodeType = LAST8
```

```
; CardDecodeType = LAST9
```

```
; CardDecodeType = MASK
```

```
mask=hhhhhhhh*
```

```
bitmask=$7FFFFFFF
```

```
[ cards. Card]
```

```
ExcludedPrefix=05015005
```

```
; CardDecodeType = ANGSTREMCARD
```

```
; CardDecodeType = ANGSTREMBRASLET
```

```
code=2
```

```
; CardDecodeType = LAST8
```

```
; CardDecodeType = LAST9
```

```
CardDecodeType = MASK
```

```
mask=ddd
```

```
; mask=hhhhhhhh*
```

```
bitmask=$FFFFFFFF
```

```
DecodeTypeEx = 0
```

```
[ cards. RegularCard]
```

```
; CardDecodeType = ANGSTREMCARD
```

```
; CardDecodeType = ANGSTREMBRASLET
```

```
code=22
```

```
[ cards. StaffCard]
```

```
; CardDecodeType = ANGSTREMCARD
```

```
; CardDecodeType = ANGSTREMBRASLET
```

```
code=21
```

```
[ cards. PDSCard]
```

```
; CardDecodeType = ANGSTREMCARD
```

```
; CardDecodeType = ANGSTREMBRASLET
```

```
code=24
```

3.1.14 Разберем по секциям

```
[ frf]
```

```
fsUnit=fsdecodecard.upas
```

Эта секция отвечает подключение файла со скриптом (FastScript). Можно написать свой алгоритм для обработки трека.

```
[ general]
```

```
usemask=1
```

```
TrackResultLog=
```

TrackResultLog= - имя файла, в котором будет результат обработки трека. (если надо делаем, обычно убираем)

usemask=1

(version.txt строка 41)

Если usemask=1, то секция [cards] игнорируется

Если usemask=1, тогда при декодировании проверяется, какой маске соответствует считанный трек.

После этого происходит обращение к секции [cards.ИМЯ МАСКИ]

Ставим единичку и создаем секцию [mask] (если нет), в ней прописываем

card_mi (произвольное имя) = (маска, по которой смотрим в какую секцию отправить на обработку трек)

В наем примере сделаем так:

```
card_mi = fare* (fare* берется из трека fare[A1B2C3D4] (....) <LF>, * - любой символ)
```

Таким образом, **все треки, которые содержат fare будут определяться и направляться на обработку в секцию card_mi**

Также в этом случае необходимо создать секцию [cards.card_mi], если её нет !

В этой секции могут содержаться различные параметры (см. version.txt), но рассмотрим под наш случай.

Нужно будет для обработки трека полученного **fare[A1B2C3D4] (....) <LF>** использовать следующие параметры: **ExcludedPrefix, code, CardDecodeType, mask**.

ExcludedPrefix-Отрезаем от трека префикс.

Для того, чтобы корректно обработать наш трек fare[**A1B2C3D4**] (....) <LF> , необходимо **удалить fare[**.

Тогда необходимо прописать в параметре **ExcludedPrefix** следующее:

ExcludedPrefix=fare[

Таким образом, мы получим внутри decodercards следующую последовательность трека **A1B2C3D4**] (....) <LF>, где жирным указано интересующая нас последовательность в шестнадцатеричной системе.

Далее используется параметр **CardDecodeType** - **этот параметр определяет алгоритм декодирования**. [Существуют несколько вариантов](#), например, LAST8, LAST9, MASK, FastScript и т.п.

В нашем случае мы используем **MASK** (накладываем маску и сравниваем с нашим треком) **CardDecodeType = MASK**

далее необходимо добавить параметр mask, который собственно, и определяет её.

Из version.txt строка 32 описание следующее

mask=**hh* - взять 3-й и 4-й символ как цифры в шестнадцатеричном формате
или
mask=**dd* - взять 3-й и 4-й символ как цифры в десятичном формате

Поскольку нам из трека **A1B2C3D4] (....) <LF>** интересна лишь последовательность A1B2C3D4, то наша маска выглядит следующим образом: **mask = hhhhhhhh**. Т.е. берем символы с 1 по 8 в шестнадцатеричном формате и преобразуем в конечный вариант в десятичный формат (DEC).

Остается для обработки добавить лишь параметр **code** - он отвечает за тип карты.
code=2 (фичи с ним на обучении только, либо сами)

На выходе при треке **A1B2C3D4** должны получить **2712847316**

3.1.15 Таким образом **получаем следующую конфигурацию decodercards.ini при TCPCardReader**

```
[ frf ]
fsUnit=fsdecodecard.upas

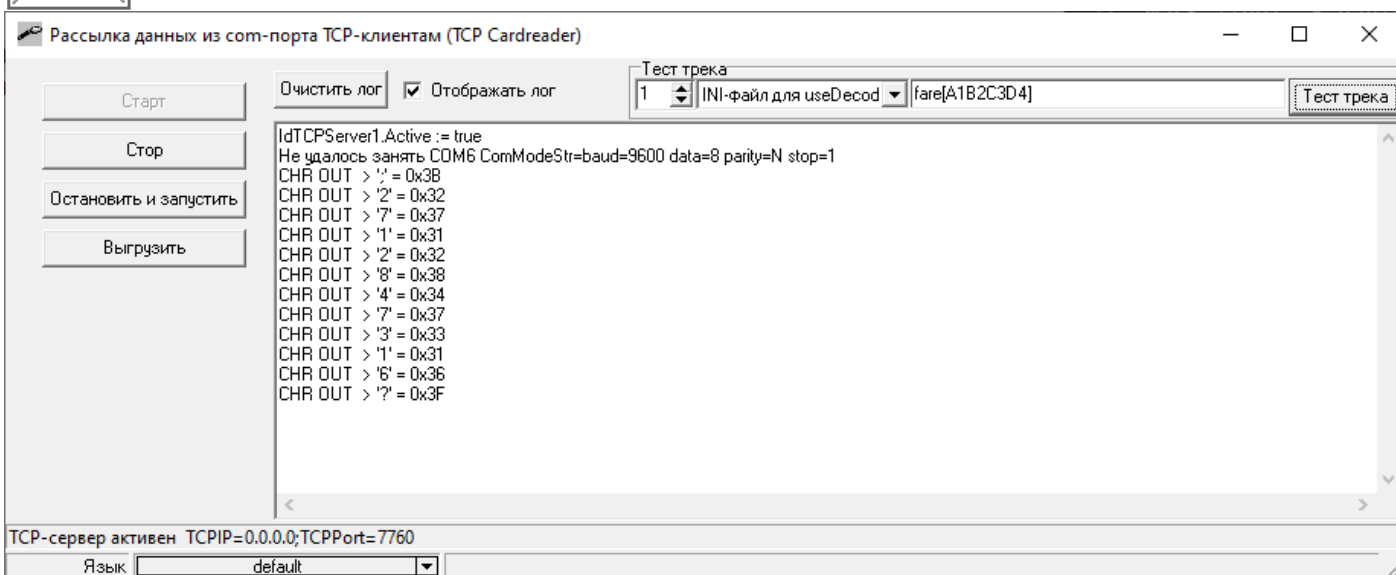
[ general ]
usemask=1

[ mask ]
    card_mi = fare*

[ cards. card_mi ]
ExcludedPrefix=fare[
code=2
CardDecodeType = MASK
mask=hhhhhhhh
```

3.1.16 Можно проверить правильность настроек **с помощью теста в TCPCardReader**. В поле слева от кнопки тест, **вводится трек, который должен передаться в decodercards.dll**

image not found or type unknown



В отображение вы должны получить последовательность отправки символов клиентам (трек).

Если вместо этого получаете :

CHR OUT> ;

CHR OUT> 0

CHR OUT> ?

Вы неправильно настроили декодирования - повторяете изучение материала с 1 пункта

3.1.16 Можно также убедиться по логам decodercards, при использовании декодирования в логам TCPCardReader можно увидеть только информацию отправляемую клиентам.

Лог правильно настроенного TCPCardReader:

```
22. 09. 2021 18: 05: 35 581059359 2C74> CALL extractcardnoexp_int64: track=fare[F2D1BF1F]
191,53746 1K (0004,08) ; term0= , term1=
22. 09. 2021 18: 05: 35 581059359 2C74> CALL extractcardnoex: track=fare[F2D1BF1F] 191,53746 1K
(0004,08) ; term0= , term1=
22. 09. 2021 18: 05: 35 581059359 2C74> UseMask2: UseMask=1
22. 09. 2021 18: 05: 35 581059359 2C74> CALL extractcardnoex2: track=fare[F2D1BF1F] 191,53746 1K
(0004,08) ; term0= , term1=
22. 09. 2021 18: 05: 35 581059375 2C74> EXIT extractcardnoex2: result=4073832223 , cardcode=2 ,
ExtractWithoutPrefix2=F2D1BF1F] 191,53746 1K (0004,08), ci=card_mi
22. 09. 2021 18: 05: 35 581059375 2C74> EXIT extractcardnoexp_int64: result=4073832223 ,
```

cardcode=2

CALL extractcardnoexp_int64: track=fare[F2D1BF1F] 191,53746 1K (0004,08) ; term0= , term1= | - **прилетел трек**

EXIT extractcardnoex2: result=4073832223 , cardcode=2 ,
ExtractWithoutPrefix2=F2D1BF1F] 191,53746 1K (0004,08), ci=card_mi | **ci - имя секции, по которой трек обрабатывается;**

ExtractWithoutPrefix2=F2D1BF1F] 191,53746 1K (0004,08) - **результат после исключения из трека префикса**, | result=4073832223 -**результат декодирования EXIT extractcardnoexp_int64: result=4073832223 , cardcode=2 | результат на выходе из decodercards.dll !!!**

3.1.17 После правильного настроенного TCPCardReader, необходимо настроить decodercards.ini при abonementmanager или при коннекторе, или при gkhostconnect.

Рассмотрим простой и стандартный вариант- это Abonementmanager

3.1.18 Открываем decodercards.ini и видим

```
[ frf]
fsUnit=fsdecodecard.upas

[ general]
usemask=1
TrackResultLog=

[ mask]
card = 05015005*
card2= 778=12345678=*
cardfile=file: //cards.txt

[ cards]
CardPrefix    = 05015005
;              05015005280507210
CardPrefix    = 750=
CardPrefix    = 778=999999999=
CardPrefix    = 778=123456789=
StaffCardPrefix      = 778=87121234=
StaffCardPrefix      = 778=201050001
```

RegularCardPrefix = 778=444444444=

PDSCardPrefix = 811q

[cards. cardREPLACECARDNO]

ExcludedPrefix=

code=2

CardDecodeType = REPLACECARDNO

NewCardNo = 5678956

[cards. ARRAYOFBYTES]

CardDecodeType = MASK

mask=*cccc*

bitmask=\$7FFFFFFF

MaskType=ARRAYOFBYTES

code=2

[cards. Card2]

ExcludedPrefix=778=12345678=

; CardDecodeType = ANGSTREMCARD

; CardDecodeType = ANGSTREMBRASLET

code=2

; CardDecodeType = LAST8

; CardDecodeType = LAST9

; CardDecodeType = MASK

mask=hhhhhhh*

bitmask=\$7FFFFFFF

[cards. Card]

ExcludedPrefix=05015005

; CardDecodeType = ANGSTREMCARD

; CardDecodeType = ANGSTREMBRASLET

code=2

; CardDecodeType = LAST8

; CardDecodeType = LAST9

CardDecodeType = MASK

mask=ddd

```

; mask=hhhhhhh*
bitmask=$FFFFFFFF

DecodeTypeEx = 0

[ cards. Regular Card]
; CardDecodeType = ANGSTREMCARD
; CardDecodeType = ANGSTREMBRASLET
code=22

[ cards. StaffCard]
; CardDecodeType = ANGSTREMCARD
; CardDecodeType = ANGSTREMBRASLET
code=21

[ cards. PDSCard]
; CardDecodeType = ANGSTREMCARD
; CardDecodeType = ANGSTREMBRASLET
code=24

```

3.1.18 Приводим к этому виду, удалив и почистив почти все

```

[ general]
usemask=1

[ mask]
card_all = *

[ cards. card_all]
code=2

[ cards. Card]
code=2

[ cards. Regular Card]
code=22

[ cards. StaffCard]
code=21

```

```
[ cards. PDSCard]
code=24
```

Подробнее

```
[ mask]
card_all = *
```

Как и по аналогии с TCPCardReader делаем настройку по маске.

card_all = * | * - все получаемые треки. Поскольку abonementmanager подключается к TCPCardRead, на стороне которого происходят все преобразования треков, получаемых со считывателей, в абонемент manager прилетает TCP **только трек в десятичном формате.** Т.е. исходя из выше обработанного к нам прилетает 2712847316. Соответственно, мы уже получили почти готовый трек.

Создаем секцию **[cards.card_all]**

И устанавливаем "физический" тип карты **code=2.**

Также для того, чтобы корректно отрабатывалось определение физического типа рекомендуется использовать во всех клиент с decodercards.dll данные секции:

```
[cards.Card] - гостевая
code=2
```

```
[cards.RegularCard] - постоянная
code=22
```

```
[cards.StaffCard] - карта персонала
code=21
```

```
[cards.PDSCard] - ПДС
code=24
```

Соответственно, **при прикреплении карты в базу будет прикрепляться трек с физическим типом 2.** При наличии служебных секций он определится, как гостевой

3.1.19 Осталось настроить декодирование при gkhostconnect

3.1.20 При использовании **GKHOST** в **GKHOSTCONNECT** на основе выше перечисленного трека прилетают карты вида **CARD A1 B2 D3 C4 !**

На конце обычно еще бывает F (*CARD A1 B2 D3 C4*FF ...) зависит от прошивки считывателя

3.1.21 Открываем decodercards.ini при gkhostconnect и удаляем все, оставляя только необходимое и приводим к следующему виду

```
[ fr f]
fsUnit=fsdecodecard. upas

[ general]
usemask=1
TrackResultLog=

[ mask]
card = *

[ cards. card]
code=2
CardDecodeType = MASK
mask=hhhhhhh
```

Поскольку к нам прилетает карта сразу HEX значении. А при передаче в decodercards элементы ##### CARD откидываются автоматически, остается лишь преобразовать hex в dec. И поставить корректный тип карты в нашем случае это 2.

4. Готово. (отходим и нервно курим в сторонке)

Дополнительные материалы

[Настройка_decodercards_.txt](#)

[version \(decodercards\).txt](#)

[version \(TCPCardReader\).txt](#)

Версия #19

Денис Сорокин создал 30 August 2021 10:38:05

Игорь Докутович обновил 6 October 2022 12:23:41