Эмулятор -картсервер **wicardd** рассмотрим а примере версии -

## Версия 1.12 (2013-02-23) (doublemu)

сначала рассмотрим общие вопросы, в конце - практические

Для работы эмулятора потребуется 3 файлика:

1 - собственно самого эмулятора – бинарник (wicardd), 2 - конфиг - файла (wicardd.conf) и

3 - скрипта запуска (wicardd\_cam.sh)

Для использования в azbox вариант эмулятора - **wicardd-mipsel**

размещается в - /usr/bin с правами 755

Основные команды (те ключи командной строки запуска):

# -h help показать помощь

# -d daemon запустить в режиме демона

# -v verbose

# -c [filename] use config filename использовать файл конфигурации отличный от /etc/wicardd.conf и ./wicardd.conf

# -l [filename] use log filename

# -a сохранить результат автоконфигурации в конфиг файле

# -b [arg] configure WiBase конфигурирование радиобазы

# -w [arg] configure WiCard конфигурирование радиозонда

# -C calibrate mode

# -A spectrum analizer mode

# -R log rssi/lqi

# -L [channel] card logger логирование протокола обмена карты и радиозонда

# -w "arg=1:arg1=:arg2"

#args:

#show показать настройки радиозонда

#devaddr=1..7 присвоить адресс радиозонду

#atrno=0..15 использовать ATR номер 0-15

#atrconf= дополнительные настройки протокола обмена карты доступа

#atrX=3B240030423030 задать значение для ATR номер X (atr0=, atr1=,... atr15=)

#freqno=0..3 использовать частотный канал 0-3

#updatefw обновить прошивку радиозонда

#calibrate калибровать частоту передатчика ()# пример: wicardd -w "devaddr=3:atrno=5:freqno=1"

# -b "arg=1:arg1=:arg2"

#args:

#show показать настройки радиобазы

#freqno=0..3 использовать частотный канал 0-3

#updatefw обновить прошивку радиобазы

#DM800 mknod -m 660 /dev/ttyS0 c 4 64

[section\_name.preset] # задает значение по умолчанию, для всех секций [section\_name]

debug = 4

# пример:

[reader.preset] # задает значение по умолчанию, для всех секций [reader]

reconnect\_delay = 100

debug = 4

[server.preset] # задает значение по умолчанию, для всех секций [server]

max\_not\_decoded = 5

debug = 4

Не всеми параметрами будем пользоваться , однако при написании скриптов запуска часть их просто необходима

## КОНФИГ - ФАЙЛ

**wicardd.conf**

Все настройки работы эмулятора собраны в нем. Используются необходимые , в зависимости от поставленной задачи.

Конфиг-файл состоит из секций.

Условно их можно разделить на группы:

* секции общего назначения
* секции устройств
* секции серверов
* секции ридеров
* секции фильтров

## Секции общего назначения

**[global]**

active = 1 # включение секции (0-выключение)

syslog = 1 # вывод информации в syslog

log = /tmp/wicardd.log # установить логгирование в файл (место нахождения файла - /tmp/wicardd.log)

Если выводить лог на комп (к примеру) , то

log\_udp = hostname:port # сетевое логирование на hostname:514

log\_size = 100 # максимальный размер лог-файла (установить в 100кб)

debug = 3 # уровень полноты вывода в лог ( 3 - полный ; 0 - выключить )

daemon = 0 #1 - запускать в режиме демона

name = test\_server # имя сервера, применяется для включения секций по маске active\_if

#@/path/file вычитать имя из файла

ecm\_cache\_size = 128 # размер ECM кеша

pending\_ecm\_cache\_size = 128 # размер буфера

account\_file = ./wicard.account # путь к файлу с описанием пользователей

account\_diff\_file = ./wicard.account.diff # путь к файлу с описанием пользователей

default\_balancer\_debug = 0

thread\_stack\_size = 128 # размер стека выделяемого потоку в килобайтах

thread\_priority = 0 # приоритет потоков 0..99

ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс, значение по умолчанию

dre\_hack # кол -во сравниваемых байт UA для индивидуальных EMM CAS DRE

ecm\_dedup\_delay = 1000 # задержка отправки ECM запроса в мс

filter\_gvar\_array\_size = 32 # размер массива глобальных переременных для фильтров

$replace\_val1 = val #

$replace\_val2 = val

## [webif] - секция веб-интерфейса

port = 8081 # - (ip номер порта ресивера)

debug = 0

active = 1

active\_if = name\* # активировать секцию если имя сервера соответствует wildcard

active\_if = !name\* # активировать секцию если имя сервера не соответствует wildcard

allow\_ip = 127.0.0.1/32, 10.0.0.1/8 # разрешить доступ только с перечисленных адресов/сетей

deny\_ip = 127.0.0.1/32, 10.0.0.1/8 # запретить доступ с перечисленных адресов/сетей

**[exmlog]**

log\_path = /tmp/ queue\_size = 10

log\_ecm = 1 # логировать ECM запросы

log\_emm = 1 # логировать EMM запросы exm\_flush\_delay = 30 # задается в секундах

ecmlog\_mode = 0 #

skip\_fakecw = 0 # не сохранять в логе запросы с фейковыми ответами

skip\_rejected = 0 # не сохранять в логе отвергнутые запросы

**[discovery]**

active = 1

debug = 0

reader\_debug = 0 # уровень логирования для создаваемых ридеров

request\_ad = 0 # запрашивать тестовые настройки с сервера wicard.tv bind = 1 #22339

**[account]** # секция конфигурации пользователей (повторяется для каждого пользователя)

# используется для разграничения доступа к серверам описанным в секции [server]

login = wicard password = wicard

auto\_update = 1 #accept AU

filter = 0,name1,name2 # использовать фильтр

provider = caid:provid,provid;caid@UA:provid@SA,provid@SA # список провайдеров, которые будут анонсированы клиенту

allow\_ip = 127.0.0.1/32, 10.0.0.1/8 # разрешить доступ только с перечисленных адресов/сетей

deny\_ip = 127.0.0.1/32, 10.0.0.1/8 # запретить доступ с перечисленных адресов/сетей

**Секции устройств**

**[tuner]** #

device = /dev/ttyS0

device = COM1 #COM-порт под Windows

#device = /dev/tts/0

baudrate = 115200 # скорость обмена (9600, 19200, 38400, 57600, 115200)

protocol = auto #auto, HSIC, SSSP, Bomba, GI, DSR9500, GS, Alpha, DSR9500old, GBox

provider = caid:provid,provid;caid@UA:provid@SA,provid@SA

# список провайдеров, которые будут анонсированы клиенту

auto\_provider = all, reader\_name, caid # автоматический импорт всех идентов с reader устройств с таким же CAID или по имени

globo = 1 # если 1 - сразу отдавать ноли в DCW для неподдерживаемых идентов (ускоряет переключение каналов)

#write\_delay = 1 #ms

#timeout = 50 #ms

debug = 3

#caid = 0500

filter = 0,name1,name2 # использовать фильтр

**[dvb]**

type = auto #auto, DVBAPI1, DVBAPI3, STAPI, OpenXCAS, DGSAPI, QBoxHD, VDR

adapter = 0 # номер адаптера

cadev\_offset = 0 #fix для Enigma на IPBox filter = 0,name1,name2 # использовать фильтр

beta\_tunnel = 0 # тунелирование запросов

decode\_try\_count = 0 # кол -во попыток декодирования канала

dre\_fix = 1 #1 - игнорировать провайдер DRE, помогает в случае если сервер возвращает provider = 0 (4ae1:0)

sdt = 1 # вычитывать названия сервисов из потока

ecm\_info = 1 # создавать файл /tmp/ecm.info

#1 - формат WiCard

#2 - формат mgcamd

twin = 0 # поддержка TWIN протокола (вычитывается SDT и NIT для определения частоты транспондера и орбитальной позиции спутника )

ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс, требуется для работы с TWIN протоколом

rerequest\_not\_decoded = 0 # отправлять повторный запрос если не декодировано

ecm\_double\_check = 1 # сравнивать предыдущую и текущую ECM и отправлять на декодирование если совпадают

auto\_update = 1 # включить AU

emm\_filter\_mode = 0 #0 - вычитывается весь EMM PID, фильтруется WiCard

#1 - вычитывать EMM PID с настроенными фильтрами для linux DVB API

debug = 3

stapi\_pmt\_map = 00010102030304 # соответствие pmt файлов к номеру демультиплексора

#pmt.tmp pmt1.tmp pmt1\_1.tmp pmt1\_2.tmp pmt2.tmp pmt2\_1.tmp pmt2\_2.tmp

#demux #0 PTI2

#demux #1 PTI

#demux #2 PTI

#demux #3 PTI1

#demux #4 PTI1

multi\_decode = 0 #1 - отправлять ECM со всех каналов текущего транспондера

#2 - отправлять ECM со всех каналов транспондеров описанных параметром dvbs

#3 - отправлять TS потоки со всех каналов транспондеров описанных параметром dvbs

multi\_decode\_demux = 0 # номер демультиплексора

multi\_decode\_resend\_ecm = 0 #0 - отправлять ECM только при смене таблицы 80/81

#1 - отправлять ECM после каждого цикла сканирования транспондера

multi\_decode\_recheck\_pat = 0 #1 - отслеживать изменение PAT и пересканировать транспондер

multi\_decode\_sid = 01,0A # список каналов, которые будут декодироваться

multi\_decode\_ignore\_sid = 01,0A # список игнорируемых каналов, не вычитывать PMT

dvbs = freq:pol:sr:lnb:diseqc #lnb - X 10750, U - Universal

dvbs2 = freq:pol:sr:fec:roll-off:mod:lnb:diseqc:filter\_group

#freq frequency in MHz

#pol H,V,R,L

#sr symbolrate

#fec 12,23,34,45,56,67,78,35,910, 0 - AUTO

#roll-off factor 35,25,20, 0 - AUTO

#mod QPSK, 8PSK

#lnb - X 10750, U - Universal

#diseqc 0 do not use, 1-16 port number

multi\_decode\_diseqc = 0 # diseqc version 0 - 1.0 (commited); 1 - 1.1(uncommited)

tune\_timeout = 500 #

multi\_decode\_scan\_timeout = # multi\_decode\_softcsa = 0

lnb\_sharing = 0

check\_cc = 0 stream\_buffer\_size = #

stream\_udp = host:port # стримить все TS на host:port

stream = sid\_hex1:host1:port # стримить привязанные к SID vpid, apid, если SID = 0, стримить все каналы stream = sid\_hex2:host2:port #

stream = sid\_hex3:host3:port #

dmx\_buffer\_size = #

dmx\_pes\_buffer\_size = #

remap\_sid\_on\_vpid = 0 #

**[card]** # секция конфигурации радиозондов (одна секция на один радиозонд)

name = card1 #

cas = irdeto # тип эмулируемой кодировки карты доступа (auto, conax, viaccess, irdeto, dre)

channel = 2 # сетевой адрес/канал радиозонда

provider = caid:provid,provid;caid@UA:provid@SA,provid@SA

# список провайдеров, которые будут анонсированы клиенту

auto\_provider = all, reader\_name,caid # автоматический импорт всех идентов с reader устройств с таким же CAID или по имени

send\_null\_ins = 1 # отправлять тюнеру команду NULL (60) в период ожидания CW

delay\_after\_ecm = 500 # дополнительная задержка после отправки ECM серверу

ascserial = 0000000000U0000000A

ACS = 0608 #Irdeto ACS

coco = MCR

dre\_version=1 #DRECrypt version 1/2

write\_delay = 0

filter = name,idx # перечень используемых фильтров

debug = 3 # уровень логирования 0 - 3

active = 0 # отключение секции

filter\_group = 0 # группа, используется в условиях фильтра

**[card.auto]** # если секция присутствует - автоматически будут добалены процессы

# обрабатывающие запросы от радиозондов не описанных в секции(ях) [card]

cas = auto channel = x debug = 3

send\_null\_ins = 1 # отравлять команду 60 ресиверу (карта занята )

delay\_after\_ecm = 400

active = 0 # отключение секции

filter\_group = 0 # группа, используется в условиях фильтра

## Секции серверов

**[server]** # конфигурация сервера

type = newcamd525 # тип сервера (**поддерживается newcamd525**)

provider = caid:provid,provid;caid@UA:provid@SA,provid@SA

# список провайдеров, которые будут анонсированы клиенту

auto\_provider = all, reader\_name, caid # автоматический импорт всех идентов с reader устройств с таким же CAID или по имени

port = 34000 # порт сервера

key = 0102030405060708091011121314 #DES ключ

debug = 3 # уровень логирования 0 - 3

active = 0 # отключение секции

ecm\_queue\_size = 8 # размер очереди запросов

extended\_newcamd = 0 # отдавать несколько CAID на одном порту

#1 - отдаем все что прописано в данной секции в provider

#2 - сканируем все сервера типа newcamd и передаем информацию прописанную в provider найденных серверов

keepalive = 300 # закрывать неактивное соединение по истечению n секунд

max\_not\_decoded = N # отправлять нулевой DW в случае N неудачных попыток декодирования

guest\_account = account\_login #

filter\_group = 0 # группа, используется в условиях фильтра

force\_auto\_update = 0 #1 - отдавать принудительно номера карт для AU

allow\_ip = 127.0.0.1/32, 10.0.0.1/8 # разрешить доступ только с перечисленных адресов/сетей

deny\_ip = 127.0.0.1/32, 10.0.0.1/8 # запретить доступ с перечисленных адресов/сетей

extra\_data = 1 #1 - отправлять расширенный статус декодирования WiCardd клиенту

**[server]** # конфигурация сервера

type = cs357x | cs378x # тип сервера **CAMD35**

provider = caid:provid,provid;caid@UA:provid@SA,provid@SA

# список провайдеров, которые будут анонсированы клиенту

auto\_provider = all, reader\_name,caid # автоматический импорт всех идентов с reader устройств с таким же CAID или по имени

port = 34000 # порт сервера

debug = 3 # уровень логирования 0 - 3

active = 0 # отключение секции

ecm\_queue\_size = 8 # размер очереди запросов

keepalive = 300 # закрывать неактивное соединение по истечению n секунд

suppress\_cmd08 = 0 #

not\_decoded\_answer\_mode = 0 # 0 - по умолчанию

# 1 - отправлять нулевой DW

# 2 - игнорировать запрос, ничего не отправлять в ответ

filter\_group = 0 # группа, используется в условиях фильтра

allow\_ip = 127.0.0.1/32, 10.0.0.1/8 # разрешить доступ только с перечисленных адресов/сетей

deny\_ip = 127.0.0.1/32, 10.0.0.1/8 # запретить доступ с перечисленных адресов/сетей

**[server]** # конфигурация сервера

type = cccam # **тип сервера CCcam**

provider = caid:provid,provid;caid@UA:provid@SA,provid@SA

# список провайдеров, которые будут анонсированы клиенту

auto\_provider = all, reader\_name, caid # автоматический импорт всех идентов с reader устройств с таким же CAID или по имени

port = 34000 # порт сервера

cc\_version = 2.0.00 # версия

cc\_build = 1892 # билд

debug = 3 # уровень логирования 0 - 3

active = 0 # отключение секции

ecm\_queue\_size = 8 # размер очереди запросов

keepalive = 300 # закрывать неактивное соединение по истечению n секунд

max\_not\_decoded = N # отправлять нулевой DW в случае N неудачных попыток декодирования

filter\_group = 0 # группа, используется в условиях фильтра

allow\_ip = 127.0.0.1/32, 10.0.0.1/8 # разрешить доступ только с перечисленных адресов/сетей

deny\_ip = 127.0.0.1/32, 10.0.0.1/8 # запретить доступ с перечисленных адресов/сетей

**Секции ридеров**

Опять же условно их можно сгруппировать в секции ридеров

* устройств
* эмуляторов
* протоколов
* прочие

**Ридеры устройств**

**[reader]**

name = # уникальное имя

type = sci # **встроенный картоприемник (WiCard)**

debug = 3 # уровень логирования 0 - 3

handle\_pts = 1 # включить поддержку переключения скорости обмена с картой доступа

emm\_process\_delay = 10 #

write\_delay = 0

force\_crypted\_cw = 1 # не проверять контрольную сумму CW

ocp = 1 #overcrypt proccessor

wait\_active = 1 # ожидать активации ридера (таймаут 13 сек)

force\_caid = 0 #DRE CAID fixup

force\_provider = 0 #DRE provider fixup

force\_ua = 0000 #DRE UA

fixup force\_ua = 0000 #Irdeto UA fixup

filter = name # перечень используемых фильтров

active = 0 # отключение секции

ecm\_queue\_size = 8 # размер очереди запросов

camkey = #Irdeto camkey

camkey\_data = #Irdeto camkey\_data

boxid = #4 байта boxid

cwekey = #NDS 26 байт CWE key

boxkey #nagra boxkey

rsamod = #nagra/conax RSA модуль

rsaexp = #

auto\_update = 1 # включить поддержку EMM запросов

emm\_cache = 1 # включить EMM кеш

ecm\_ttl = 5000 # время жизни ECM запроса в мс

block\_subs\_emm = 0 # блокирование EMM с подпиской

exec\_cmd\_file = ./card.cmd # путь к файлу с коммандами, которые будут отправлены на карту после инициализации

# для DRE 59 xx xx xx CS, контрольная сумма пересчитываеся

skip\_geo = 0 # не отправлять геокод (Viaccess)

ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс

recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW

**[reader]**

name = # уникальное имя

type = tuxbox # **встроенный картоприемник DreamBox**

device = /dev/sci0

debug = 3 # уровень логирования 0 - 3

handle\_pts = 1 # включить поддержку переключения скорости обмена с картой доступа

emm\_process\_delay = 10 #

write\_delay = 0

filter = name # перечень используемых фильтров

active = 0 # отключение секции

ecm\_queue\_size = 8 # размер очереди запросов

camkey = #Irdeto camkey

camkey\_data = #Irdeto camkey\_data

boxid = #NDS 4 байта boxid

cwekey = #NDS 26 байт CWE key

aeskey = #AES ключ

emm\_cache = 1 # включить EMM кеш

ecm\_ttl = 5000 # время жизни ECM запроса в мс

skip\_geo = 0 # не отправлять геокод (Viaccess)

ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс

recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW

**[reader]**

name = # уникальное имя

type = stapi # **встроенный картоприемник ST40**

device = SC0 # картоприемник SC0, SC1

debug = 3 # уровень логирования 0 - 3

write\_delay = 0

filter = name # перечень используемых фильтров

active = 0 # отключение секции

ecm\_queue\_size = 8 # размер очереди запросов

camkey = #Irdeto camkey

camkey\_data = #Irdeto camkey\_data

boxid = #NDS 4 байта boxid

cwekey = #NDS 26 байт CWE key

aeskey = #AES ключ

emm\_cache = 1 # включить EMM кеш

ecm\_ttl = 5000 # время жизни ECM запроса в мс

skip\_geo = 0 # не отправлять геокод (Viaccess)

ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс

recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW

**[reader]**

name = # уникальное имя

type = openxcas # **встроенный картоприемник AzBox**

device = /dev/scard # номер картоприемника

debug = 3 # уровень логирования 0 - 3

handle\_pts = 1 # включить поддержку переключения скорости обмена с картой доступа

emm\_process\_delay = 10 #

write\_delay = 0

filter = name # перечень используемых фильтров

active = 0 # отключение секции

ecm\_queue\_size = 8 # размер очереди запросов

camkey = #Irdeto camkey

camkey\_data = #Irdeto camkey\_data

boxid = #NDS 4 байта boxid

cwekey = #NDS 26 байт CWE key

aeskey = #AES ключ

emm\_cache = 1 # включить EMM кеш

ecm\_ttl = 5000 # время жизни ECM запроса в мс

skip\_geo = 0 # не отправлять геокод (Viaccess)

ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс

recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW

**[reader]**

name = # уникальное имя type = phoenix # **phoenix**

device = /dev/ttyS0 #COM-порт

device = COM1 #COM-порт под Windows

intrts = 1 #RTS управляется через контроллер (для внутреннего порта D-Link)

t14 = 1 #Irdeto protocol

timeout = 1000 #

debug = 3 # уровень логирования 0 - 3

handle\_pts = 1 # включить поддержку переключения скорости обмена с картой доступа

card\_detect = 1 # определять наличии карты по сигналу CTS

clock\_freq = 3571200 # частота клока картоприемника в Гц

filter = name # перечень используемых фильтров

active = 0 # отключение секции

ecm\_queue\_size = 8 # размер очереди запросов

camkey = #Irdeto camkey

camkey\_data = #Irdeto camkey\_data

boxid = #NDS 4 байта boxid

cwekey = #NDS 26 байт CWE key

aeskey = #AES ключ

emm\_cache = 1 # включить EMM кеш

ecm\_ttl = 5000 # время жизни ECM запроса в мс

skip\_geo = 0 # не отправлять геокод (Viaccess)

ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс

recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW

**[reader]**

name = # уникальное имя

type = tuner # использовать **тюнер с установленной картой доступа как источник**

device = /dev/ttyS0 #COM-порт

device = COM1 #COM-порт под Windows

baudrate = 115200 # скорость обмена

protocol = bomba # протокол (HSIC, Bomba, DSR9500, Alpha)

provider = caid:provid,provid;caid@UA:provid@SA,provid@SA # список обслуживаемых провайдеров

write\_delay = 0

timeout = 50 # таймаут чтения данных

debug = 3

active = 0 # отключение секции

ecm\_queue\_size = 8 # размер очереди запросов

ecm\_ttl = 5000 # время жизни ECM запроса в мс

ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс

recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW

**[reader]**

name = # уникальное имя type = drecas #

device = /dev/ttyS0 #COM-порт

device = COM1 #COM-порт под Windows

dre\_cas\_type = 3 # 1 - модуль первой версии (ZM\_2.04) (2-5V, 10-RxD(компьютер TxD), 12-TxD(компьютер RxD), 13-GND)

# сигнал DTR используется для переключения между MSP и ATMEL, RTS - сброс модуля.

#

# 3 - проприетарный картоприемник GS9300HD (1-5V, 2-RxD(компьютер TxD), 3-TxD(компьютер RxD), 4-GND)

# сигналы расписаны поотношению к модулю и картоприемнику, RxD - входной сигнал для модуля, TxD - выход

# Следовательно схема должна выглядеть примерно так:

# PC\_TxD -> MAX232 -> MOD\_RxD

# PC\_RxD < - MAX232 < - MOD\_TxD

# GND ------------------- GND

emm\_key\_file = ./ # файл в котором будут сохраняться ключи DRE при работе с модулем первой версии

timeout = 2000 #

debug = 3 #

write\_delay = 1 #

ecm\_queue\_size = 8 # размер очереди запросов

ecm\_ttl = 5000 # время жизни ECM запроса в мс

ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс

recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW

## Ридеры эмуляторов

**[reader]**

name = # уникальное имя

type = goldcard | funcard # **поддерживается эмуляция 16F84 и 90S8515**

fw\_file = ./fw.hex # путь к файлу прошивки

intee\_file = ./int.hex # путь к файлу данных внутренней EEPROM (необязательный параметр)

extee\_file = ./ext.hex # путь к файлу данных внешней EEPROM (необязательный параметр)

extee\_size = 16 # размер внешней EEPROM в килобитах (необязательный параметр)

guardtime = 1 #extra guard time задержка в ETU между посылкой байт в " карту"

## [reader]

name = # уникальное имя

type = emuvia # **эмулятор viaccess**

debug = 3 # уровень логирования 0 - 3

file = ./SoftCam.Key # файл с ключами active = 0 # отключение секции

ecm\_queue\_size = 8 # размер очереди запросов

ecm\_ttl = 5000 # время жизни ECM запроса в мс

ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс

recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW

## [reader]

name = # уникальное имя

type = emucworks # **эмулятор CryptoWorks**

debug = 3 # уровень логирования 0 - 3

file = ./SoftCam.Key # файл с ключами active = 0 # отключение секции

ecm\_queue\_size = 8 # размер очереди запросов

ecm\_ttl = 5000 # время жизни ECM запроса в мс

ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс

recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW

## [reader]

name = # уникальное имя

type = emubiss # **поддежка BISS**

debug = 3 # уровень логирования 0 - 3

file = ./SoftCam.Key # файл с ключами

active = 0 # отключение секции

ecm\_queue\_size = 8 # размер очереди запросов

ecm\_ttl = 5000 # время жизни ECM запроса в мс

ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс

recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW

## [reader]

name = # уникальное имя

type = emunds # **эмулятор videoguard**

debug = 3 # уровень логирования 0 - 3

file = ./SoftCam.Key # файл с ключами

active = 0 # отключение секции

ecm\_queue\_size = 8 # размер очереди запросов

ecm\_ttl = 5000 # время жизни ECM запроса в мс

ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс

recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW

## [reader]

name = # уникальное имя

type = emuirdeto # **эмулятор irdeto2**

debug = 3 # уровень логирования 0 - 3

file = ./SoftCam.Key # файл с ключами

active = 0 # отключение секции

ecm\_queue\_size = 8 # размер очереди запросов

ecm\_ttl = 5000 # время жизни ECM запроса в мс

ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс

recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW

## Ридеры протоколов

**[reader]**

name = # уникальное имя

type = newcamd525 # тип - **клиент newcamd525**

account = login:password@host:15000:010203040506070809101112 1314

filter = name # перечень используемых фильтров

debug = 3

reconnect\_delay = 10 # в секундах

keepalive = 0 # в секундах/0 - отключить

active = 0 # отключение секции

ecm\_queue\_size = 8 # размер очереди запросов

repeat\_section = 1 # вычитать секцию N раз

auto\_update = 1 # включить поддержку EMM запросов

|  |
| --- |
| emm\_cache = 1 # включить EMM кеш |
| ecm\_ttl = 5000 # время жизни ECM запроса в мс |
| async\_mode = 1 # включить/выключить асинхронный режим (запросы передаются без ожидания ответа от сервера) |
|
| ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс |
|
| recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW |
|
| reconnect\_to\_account\_ip = name # подключаться к серверу, используя IP адрес подключенного пользователя |
| force\_auto\_update = 0 # отправлять все ЕММ с CAID:provid не учитывая UA/SA |
| # |
| extra\_data = 1 #1 - отправлять расширенную информацию о клиенте серверу |
| **[reader]** |
| name = # уникальное имя |
| type = cs357x | cs378x # тип - **клиент CAMD35** |
| provider = caid:provid,provid;caid@UA:provid@SA,provid@SA # список обслуживаемых провайдеров |
| account = login:password@host:15000 |
| filter = name # перечень используемых фильтров |
| debug = 3 |
| reconnect\_delay = 10 # в секундах |
| active = 0 # отключение секции |
| ecm\_queue\_size = 8 # размер очереди запросов |
| repeat\_section = 1 # вычитать секцию N раз |
| auto\_update = 1 # включить поддержку EMM запросов |
| emm\_cache = 1 # включить EMM кеш |
| timeout = 1500 # время ожидания ответа от сервера |
| retry\_count = 1 # кол -во попыток |
| ecm\_ttl = 5000 # время жизни ECM запроса в мс |
| ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс |
| recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW |
| **[reader]** |
| name = # уникальное имя |
| type = twin # **TWIN протокол** |
| device = /dev/ttyS0 # COM-порт |
| device = COM1 #COM-порт под Windows |
| baudrate = 115200 # скорость обмена |
| timeout = 2000 # |
| debug = 3 # |
| write\_delay = 1 # |
| ecm\_queue\_size = 8 # размер очереди запросов |
| ecm\_ttl = 5000 # время жизни ECM запроса в мс |
| ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс |
| recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW |
| file = ./freq\_table.txt # файл в котором прописывается связь между CAID:ProvID/SID и частотами транспондера (не |
| нужен при работе с DVB) |
| # формат файла: |
| #CAID:ProvID/SID,орбитальная позиция \* 10, частота транспондера в МГц |
| #0500:023700/2B00,360,12322 |
|  |
| **Ридеры прочие** |
| **[reader]** |
| type = ocp # **OverCryptProcessor** |
| ecm\_ttl = 5000 # время жизни ECM запроса в мс |
| ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс |
| recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW |
| **[reader]** |
| name = # уникальное имя |
| type = cache # **кеш ридер**, все запросы попавшие в очередь этого ридера отрабатываются только из кеша |
| debug = 3 # уровень логирования 0 - 3 |
| active = 0 # отключение секции |
| ecm\_queue\_size = 8 # размер очереди запросов |
| ecm\_ttl = 5000 # время жизни ECM запроса в мс |
| ecm\_delay = 0 # принудительная задержка отправки ECM запроса в мс |
| recalc\_cw\_checksum = 0 # пересчитать контрольную сумму СW |
| provider = caid:provid,provid;caid@UA:provid@SA,provid@SA # список обслуживаемых провайдеров |
|  |
| **Секции фильтров** |
| **[filter]** |
| type = pre|post|both|emm |
| rule = [условие ];[условие ];[присвоение];[присвоение];[функция ];[условие ];[действие] |
| debug = 0 |
| Примерчик |
| [filter] |
| name = name |
| debug = 3 |
| rule = caid=500;ident=030600;reject |
| rule = caid=500;ident=023700;accept |
| rule = caid=4ae0;ident=0;caid:=4ae1;ident:=11;accept |
| rule = fakecw |
| **Синтаксис команд:** |
| операторы присвоения: |
| := присвоить |
| += прибавить |
| -= вычесть |
| /= разделить |



%= модуль

\*= умножить

|= ИЛИ

&= И

^= исключающее ИЛИ

операторы отношения:

= равны

!= не равны

>= больше или равно

<= меньше или равно

&&= истенно если результат (A & B) не равен нулю

[] индексация данных начиная с 0, адрес задается в десятичной системе ecm[0] = 80,81 переменные:

gvar[] глобальные, доступны для всех фильтров

lvar[] локальные , привязаны к цепочке фильтров секций account, dvb, tuner

rvar[] переменные запроса, доступны для всех фильтров через которые проходит запрос

EMM

emm[] доступ к содержимому EMM, emm[0] - номер таблицы, emm[2] - младший байт размера EMM caid

pid rlen

ip ip = 127.0.0.1/32,10.0.0.1/8 проверить принадлежность к IP адресу service\_type тип:

SERVICE\_TYPE\_NEWCAMD = 4

SERVICE\_TYPE\_CS357 = 5

SERVICE\_TYPE\_CS378 = 6

SERVICE\_TYPE\_CCCAM = 7

service\_id номер порта сервера на который пришел запрос timeframe

rvar[]

lvar[]

gvar[]

client имя клиента отправившего запрос client=DVB\*,test\_account,tuner\*

dump вывод содержимого ECM/EMM в лог reset

accept reject next skip

ECM pre/post/both caid

provid ident sid pid chid group

demux

status : ECM\_REQ = 0, ECM\_PENDING = 1,

ECM\_DECODED = 2,

ECM\_DECODED\_CACHE = 3,

ECM\_DECODED\_FAKE = 4,

ECM\_NOT\_DECODED = 5,

ECM\_REJECTED = 6,

ECM\_DECODE\_ERROR = 7,

ECM\_TIMEOUT = 8

ecm\_delay service\_type service\_id ecm[]

rlen class

overcrypt banned limit

geo timeframe ip

rvar[]

lvar[]

gvar[] client betawrap betaunwrap

hash вывести в лог хеш сигнатуру запроса ban

dump reset accept reject next skip fakecw

источником запросов являются server/account, dvb, tuner

список pre/post/both/emm фильтров задаются ключем filter в секциях конфигурации account, dvb, tuner, reader

да забыл- еще балансер [balancer]

name = name # уникальное имя

name = default # в случае переопределения настроек для балансировщика, который используется поумолчанию chain = 0,1;2;3 # цепочка ридеров задается имя ридера

#chain = card\_1, сard\_2; card\_3

# что означает - запрос будет сбалансирован (по степени загрузки ридера) между двумя ридерами card\_1 и card\_2 и в случае ,

# если не будет декодирован, то будет передан в следующую по цепочке группу в которой только один ридер card\_3

mode = 0 #0 - выбирать ридер с наименьшим временем декодирования запроса (кол -во запросов в очереди \* среднее время декодирования)

#1 - выбирать ридер с наименьшим кол -вом запросов в очереди

#2 - выбирать первый ридер в группе

#3 - round-robin

# chain = cardA/3, сardB/5, cardC/123

# через косую черту задается кол -во запросов , которые можно отдать в ридер, если порядковый номер поступившего запроса больше,

# то запрос отправляется в следующий по цепочке ридер, если это последний ридер и кол -во запросов превышено,

# обнуляются счетчики запросов всех ридеров и запрос передается в начало цепочки

#4 - параллельное декодирование несколькими ридерами

debug = 3

Вот практически и все. Все это многообразие настроек показывает нам возможность очень гибкого использования wicardd. Для нас практическое применение имеют существенно меньшее количество конфигов.

Как все это использовать на практике , применительно к azbox рассмотрим далее.

Ну вот почитали теорию теперь к практике.

Что бы смотреть ТВ с использованием данного эму нам осталось сделать 2 вещи 1 запустить эмулятор (вкл - выкл его в системе по усмотрению)

2 обеспечить его запуск те подготовить рабочий конфиг

Пока будем считать что эмулятор у нас запускается (хотя без конфига – это врядли)

Начнем с него

Для начала мы должны определиться – что мы хотим

К примеру я хочу смотреть :

* бисы
* криптоверк
* шару

ну и конечно на шару 3к (без интернета)

Приступим

Для начала возьмем:

Из секций общего назначения секции

## [global]

**active = 1** # сделаем его активным

**log = /tmp/wicardd.log** # включим логгирование в файл –для анализа, потом его можно отключить

**log\_size = 100** # установим макс размер формируемого логфайла

**debug = 3** # определим полноту выводимой информации в лог файл

**daemon = 0** # оставим пока так , хотя как будет видно далее запуск эмулятора будет все равно проходить в режиме demon

В готовом конфиге коментарии можно будет убрать

Кто привык пользоваться программой syslog может подключить этот режим здесь По этой секции вроде все, далее

## [webif]

## port = 8081

**debug = 0**

**active = 1**

Не знаю, нужен ли он мне , или нет

Пока оставим, хотя я им так и не воспользовался Из секции устройств возьмем секцию

## [dvb]

**active = 1**# сделаем его активным

## type = DVBAPI3

**rerequest\_not\_decoded = 1**# если не декодировано – отправить повторный запрос

**sdt = 1**# вычитывать название сервисов из потока

**ecm\_info = 1**# создадим файл ecm.info в формате wicard в /tmp

**debug = 3**# определим полноту выводимой информации в лог файл

DVBAPI3 –прикладнной программный интерфейс для цифрового телевещания (DVB)

Для azbox type = DVBAPI3, если кто не знает какой тип использовать – можно поставить auto, эмулятор сам определит тип и его подключит

ecm\_info – нужный файлик, в нем будет находиться служебная информация (в мультиплексированном потоке условного допуска цифрового телевещания имеется 2 типа потока ECM (Entitlement Control Messages) и EMM (Entitlement Management Messages). Если вкратце, ECM управляют непосредственно возможностью просмотра программ, а EMM управляют подписками абонентов) Эта служебная информация в дальнейшем может использоваться по усмотрению (например – вывод информации на экран в различных скинах).

Все переходим к ридерам (секциям вычитки нужной нам информации )

Смотрим - что мы имеем

## [reader]

## active = 1

**type = emubiss**

**file = /usr/keys/SoftCam.Key debug = 3**

**[reader]**

**active = 1**

**type = emucworks**

**file = /usr/keys/SoftCam.Key debug = 3**

ну тут вроде все понятно, просто указываем путь размещения SoftCam.Key Пробуем запустить

займемся другим ридером

Ну это наверное самый интересный ридер, позволяющий смотреть «как хочу» пакет 3кол

В Викарде реализована поддержка эмуляции голд, фан, блу, силвер карт (это то что сегодня работает) Но принцип один

1. - нужны прошивки процессора и
2. - прошивки памяти (внешней, внутренней)
3. - указать в ридере пути их размещения

Да, для карточек помощнее PIC ов (голды) возникает уже необходимость вводить параметр временной задержки посылки байта в карту

Я взял голд карту (прошивку) и собрал такой ридер

## [reader]

## active = 1

## name = 3k

**type = goldcard**

**fw\_file = /etc/tuxbox/config/pic.hex**

**extee\_file = /usr/keys/1/ee.bin**

**debug = 3**

кстати в последней прошивке голда автообновление уже включено включать его оттдельной строкой нет необходимости

Вставляем этот ридер в наш конфиг и смотрим

ну что ж из задуманного осталость разобраться только с шарой

Идем к провайдеру и смотрим параметры просмотра купленной шары У меня VIP пакет

Для построения конфига сейчас я возьму только один пакет – плюсов Остальные возможно рассмотрим при рассмотрении фильтров

Итак

Этот пакет (НТВ+ Россия и Украина) провайдер представляет

по протоколу newcamd525 - те я – клиент протокола newcamd525

CAID:0500 IDENT:040620,040600 Порт:30000

Берем подходящую секцию (из секции протоколов)

## [reader] active = 1

**name = NTV+#это чтоб я знал**

**type = newcamd525**

**account = логин :пароль@ip адрес шары:30000:0102030405060708091011121314**

**debug = 0**

**reconnect\_delay = 10**

**emm\_cache = 1 #включим ЕММ кеш**

**ecm\_ttl = 5000**

если по этому порту транслируется еще другой пакет – ничего страшного значит ридер будет “ вычитывать ” два пакета

те на каждый порт – свой ридер

Теперь поговорим о фильтрах

Если Вы обратили внимание то построенный нами конфиг не содержит ни одного фильтра, тем не менее все работает и показывает.

Основное назначение фильтров – организовать тонкую настройку работы Викарда Структуру и синтаксис мы рассматривали ранее

Вообще общее правило таково – для каждого ридера свой фильтр К примеру, возьмем секцию бисов

[reader] active = 1

type = emubiss

file = /usr/keys/SoftCam.Key debug = 3

чтобы ридер вычитывал из потока только бисовые каналы (они имеют caid=2600), а остальные пропускал и вычитанные каналы декодировались используя SoftCam.Key

допустима такая организация

[filter]

name = biss

rule = caid=2600;accept

[reader] active = 1

type = emubiss

file = /usr/keys/SoftCam.Key debug = 3

filter = biss

В последней редакции Викарда фильтрация чуть изменена

Правила фильтрации применяются к самому DVB модулю те организуются в секции [dvb] таким образом (на примере фильтрации HD и SD при их трансляции с одного порта)

[dvb]

active = 1

type = DVBAPI3

filter = dvb

rerequest\_not\_decoded = 1

sdt = 1

ecm\_info = 1

debug = 3

[filter]

name = dvb

rule = caid=4ae1;ident=11;ident:=0;accept=3к

rule = caid=0500;ident=040600;accept=NTV+

rule = caid=0500;ident=050B00;sid=2F45,2F46,2F47,2F48,2F4 9,2F4A,3715,3716,3717,3718,3719,371A,4E85,4E86,4E8 7,4E88,4E89,4E8A;accept=NTV+

rule = reject

Что бы это значило

rule = caid=0500;ident=040600;accept=ntvplus

правило обработки (фильтрации) такое = из потока с caid=0500 получить только информацию провайдера с идентом 040600 и это правило (информацию) передать ридеру имеющему

name = NTV+

Для 040600 нет необходимости сегодня перечислять сиды Дело в том, что на HD каналах нет теперь идента 040600. Поэтому, когда включаешь HD канал (сид канала), то и выбирать не приходится, каким идентом открывать канал. А вот когда включаешь SD канал, то там присутствуют оба идента.

И как раз фильтр для 040620 перечисляет, каким только каналам можно открываться данным идентом.

rule = caid=0500;ident=040620;sid=2F45,2F46,2F47,2F48,2F4 9,2F4A,3715,3716,3717,3718,3719,371A,4E85,4E86,4E8 7,4E88,4E89,4E8A;accept= NTV+

В конце фильтра rule = reject, а не accept

Таким образом запрещаются все отстальные каиды и иденты, которые ранее не перечислены в фильтре.

А если поставить accept, то не перечисленный каид или идент будет проходить через такой фильтр. И Викард будет " тратить" время и ресурс в поисках, что с ним делать.

rule = caid=4ae1;ident=11;ident:=0;accept=3к ident:=0 - это значит , что присваивается нулю.

Те если для (4AE1:11) - то заменить id провайдера на 0 (ident:=0) и передать запрос в диспетчер запросов .

Подмена нужна только в том случае , если ключи берутся с хитрого сервера (т.е. пользуемся шарой). Это основные момента по фильтрации

Итак подошло время рассмотреть организацию запуска эмулятора

Для начала необходимо определить - Куда мы положим конфиг файл Смотрим описание викарда (его команды )

Основные команды (те ключи командной строки запуска):

## # -h help показать помощь

**# -d daemon запустить в режиме демона**

**# -c [filename] use config filename использовать файл конфигурации отличный от /etc/wicardd.conf и**

**./wicardd.conf**

Рекомендуемый вариант запуска (от автора)

## /usr/bin/wicardd -mipsel –d

Однако в этом случае эмулятор сам автоматом ищет конфиг файл в /etc/wicardd.conf или в текущей директории

./wicardd.conf

Тем не менее при запуске эмулятора без конфиг файла появится сообщение в логе что не найден конфиг файл

/var/tuxbox/config/wicardd.conf

Таким образом можно положить конфиг файл в: 1 - /etc/wicardd.conf

1. - /var/tuxbox/config/wicardd.conf Если нет таких папок - создать Конфигу обычно права 644

Самим папкам (если их создали) - 755

1. - Есть еще вариант – использовать в командной строке запуска параметр –c

Использование его позволяет разместить конфиг файл в любом удобном месте, однако в командной строке после него необходимо указать полный путь размещения конфига

1. - Наконец можно создать сим линк с каталога var на etc/ tuxbox Итак общий cкрипт запуска выглядит так

## /usr/bin/wicardd -mipsel -d -c /etc/tuxbox/config/wicardd.conf &

Разобрались с конфигом Теперь – как запустить

Напишу чуть позже, пока же кому не терпится проверить все это предлагаю простой вариант

Это тот случай если Вы будете использовать только один этот эмулятор, не использовать камд менеджер

В этом случае при запуске ресивера викард загрузится и постоянно будет Вами использоваться Запуск викарда – с запуском ресивера

Берем приложенный ниже скриптик. Кладем его в /etc/rcS.d

права 755 запуск и вперед