Author: rupp,scd
Project: Michigan Game Machine
Date: 01/06/13

ANALYSIS OF SMARTCARD WORKFLOW

Table of Contents

1. WORKFLOW

comments follows the APDU commands in blue.

BEGIN SESSION

SCardTransmit (handle 0xEA010000):
transmitted:
80 A4 00 00 02 FF 02

(ISO7816-4) Select the file , selection by file identifier, first record
File identifier = 0xFF02

SCardTransmit (handle 0xEA010000):
transmitted:
80 B2 01 00 04
received:
57 54 47 4C 90 00 (= “WTGL”)

(ISO7816-4) read the first record of the selected file , expects 4 bytes of data
==> receive content of first record= 0x57 0x54 0x 47 0x4C

SCardTransmit (handle 0xEA010000):
transmitted:
80 84 00 00 08
received:
36 25 90 6C A2 CB E7 E4 90 00

ACOS3-specific command to generate a 8 byte random number, not ISO command.
generate 8 byte random number. Usually this number is going to be used as common challenge
ACOS3 START SESSION
read a random number from card and start mutual authentication process to generate the session key K_S.

==> receive the 8 byte card random number: 0x36 0x25 0x90 0x6C 0xA2 0xCB 0xE7 0xE4

SCardTransmit (handle 0xEA010000):
transmitted:
80 82 00 00 10 B3 31 C9 E3 0D FF E4 10 38 EE 46 94 27 4B A6 A5
received:
61 08

BASED ON ISO 7816-8 MUTUAL AUTHENTICATION ON data 0xB3 0x31 0xC9 0xE3 0x0D 0xFF 0xE4 0x10 0x38 0xEE 0x46 0x94 0x27 0x4B 0xA6 0xA5

ACOS3 AUTHENTICATE.
Encrypt the 8 bytes card random number with the terminal key K_T.

0xB3 0x31 0xC9 0xE3 0x0D 0xFF 0xE4 0x10= DES(0x36 0x25 0x90 0x6C 0xA2 0xCB 0xE7 0xE4 , K_T)

sends the 8 bytes terminal random number **0x38 0xEE 0x46 0x94 0x27 0x4B 0xA6 0xA5**

==> receive: retrieve 8 bytes of answers for the mutual authentication

SCardTransmit (handle 0xEA010000):
transmitted:
80 C0 00 00 08
received:
1B D6 90 97 1C 3C D3 DF 90 00

ISO7816-4/ACOS3 GET RESPONSE

card answers 0x1B 0xD6 0x90 0x97 0x1C 0x3C 0xD3 0xDF to mutual authentication request on given submitted data

this is the terminal random number encrypted with the key session K_S. (which has been computed at this step)

SCardTransmit (handle 0xEA010000):
transmitted:
80 20 07 00 08 FF 84 8E 82 0A 2E A6 00
received:
90 00

----------> SECURE CHANNEL NOT USED (COMMUNICATIONS ARE NOT

CIPHERED ?) BY DES with sessions keys K_S ?

PROPRIETARY ISO 7816-4 VERIFY command .
ACOS3 SUBMIT CODE command

SCardTransmit (handle 0xEA010000):
 transmitted:
  80 20 01 00 08 20 1E 97 7E 04 AD A1 AD
 received:
  90 00

SECRET number 1 (CODE REFERENCE ACS1)
(8 bytes code or 8 bytes encrypted with session keys) : 20 1E 97 7E 04 AD A1 AD

SCardTransmit (handle 0xEA010000):
 transmitted:
  80 A4 00 00 02 F0 01
 received:
  91 01

SELECT FILE id=0xF0 01

SCardTransmit (handle 0xEA010000):
 transmitted:
  80 B2 00 00 04
 received:
  04 00 00 00 90 00

read record number 0 of previously selected file
==> 04 00 00 00

SCardTransmit (handle 0xEA010000):
 transmitted:
  80 B2 01 00 04
 received:
  04 00 00 00 90 00

read record number 1 of previously selected file
==> 04 00 00 00

SCardTransmit (handle 0xEA010000):
 transmitted:
  80 B2 02 00 04
 received:
  02 00 00 00 90 00
read record number 2 of previously selected file
==> 02 00 00 00

SCardTransmit (handle 0xEA010000):
transmitted:
80 A4 00 00 02 F0 01
received:
91 01

SELECT FILE id=0xF0 01

SCardTransmit (handle 0xEA010000):
transmitted:
80 B2 07 00 04
received:
08 02 00 00 90 00

read record number 7 of previously selected file
==> 08 02 00 00

SCardTransmit (handle 0xEA010000):
transmitted:
80 A4 00 00 02 F0 02
received:
91 02

SELECT FILE id=0xF0 02

SCardTransmit (handle 0xEA010000):
transmitted:
80 B2 00 00 F8
received:
47 1E 00 00 55 06 00 00 41 41 00 02 E1 28 19 00 42 42 00 02 E1 28 19 00 42 4C
46 02 E1 28 19 00 43 4B 00 02 E1 28 19 00 43 4C 43 02 E1 28 19 00 43 4C 44 02
E1 28 19 00 43 4E 4C 02 E1 28 19 00 43 52 43 02 E1 28 19 00 46 4C 46 02 E1 28
19 00 46 54 53 02 E1 28 19 00 47 49 47 02 E1 28 19 00 47 53 44 02 E1 28 19 00
4A 4F 42 02 E1 28 19 00 4C 42 00 02 E1 28 19 00 4C 4E 46 02 E1 28 19 00 4D 4D
00 02 E1 28 19 00 4D 4E 52 02 E1 28 19 00 4D 54 4D 02 E1 28 19 00 50 45 4E 02
E1 28 19 00 52 48 44 02 E1 28 19 00 52 4E 52 02 E1 28 19 00 52 4E 57 02 E1 28
19 00 52 52 00 02 E1 28 19 00 53 42 4B 02 E1 28 19 00 53 43 00 02 E1 28 19 00
53 4F 53 02 E1 28 19 00 54 4A 4D 02 E1 28 19 00 55 42 4B 02 E1 28 19 00 56 43
4C 02 E1 28 19 00 57 54 52 02 E1 28 19 00 90 00

read record number 00 of previously selected file
==> 47 1E 00 … 28 19 00 (EITHER CLEAR OR DES CIPHERED DATA)

SCardTransmit (handle 0xEA010000):
transmitted:
80 D2 00 00 F8 47 1E 00 00 56 06 00 00 41 41 00 02 8A 26 19 00 42 42 00 02 8A
26 19 00 42 4C 46 02 8A 26 19 00 43 4B 00 02 8A 26 19 00 43 4C 43 02 8A 26 19
00 43 4C 44 02 8A 26 19 00 43 4E 4C 02 8A 26 19 00 43 52 43 02 8A 26 19 00 46

4C 46 02 8A 26 19 00 46 54 53 02 8A 26 19 00 47 49 47 02 8A 26 19 00 47 53 44
02 8A 26 19 00 4A 4F 42 02 8A 26 19 00 4C 42 00 02 8A 26 19 00 4C 4E 46 02 8A
26 19 00 4D 4D 00 02 8A 26 19 00 4D 4E 52 02 8A 26 19 00 4D 54 4D 02 8A 26 19
00 50 45 4E 02 8A 26 19 00 52 48 44 02 8A 26 19 00 52 4E 52 02 8A 26 19 00 52
4E 57 02 8A 26 19 00 52 52 00 02 8A 26 19 00 53 42 4B 02 8A 26 19 00 53 43 00
02 8A 26 19 00 53 4F 53 02 8A 26 19 00 54 4A 4D 02 8A 26 19 00 55 42 4B 02 8A
26 19 00 56 43 4C 02 8A 26 19 00 57 54 52 02 8A 26 19 00
received:
90 00

write record number 00 of previously selected file with data 47 1E 00 … 26 19 00
The content of the record is been updated with new data created from the previous data and some kind of iteration (55->56) etc...

SCardTransmit (handle 0xEA010000):
transmitted:
80 B2 01 00 F8
received:
47 00 00 00 55 06 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 90 00

read data on record number 1 of selected file

answer: 47 00 00 … 00 00 00

SCardTransmit (handle 0xEA010000):
transmitted:
80 D2 01 00 F8 47 00 00 00 56 06 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
received:
90 00

write data on record number 1 of selected file 47 00 00 … 00 00 00

the data is the same except a counter that has been iterated: 55->56

SCardTransmit (handle 0xEA010000):
 transmitted:
  80 A4 00 00 02 F0 01
 received:
  99 87

select file id F0 01

SCardTransmit (handle 0xEA020000):
 transmitted:
  80 A4 00 00 02 FF 02
 received:
  90 00

select file id FF 02

SCardTransmit (handle 0xEA020000):
 transmitted:
  80 B2 01 00 04
 received:
  57 54 47 4C 90 00

read record number 1 of selected file

SCardTransmit (handle 0xEA020000):
 transmitted:
  80 84 00 00 08
 received:
  04 63 E3 9E D0 5A A7 F5 90 00

generate a 8 byte random number (same steps as previously)

SCardTransmit (handle 0xEA020000):
 transmitted:
  80 82 00 00 10 3B 13 FE 01 36 70 BD 03 58 D6 80 4B AF 4B 29 CF
 received:
  61 08

ACOS3 authenticate

SCardTransmit (handle 0xEA020000):
 transmitted:
  80 C0 00 00 08
 received:

34 9B 04 CC D0 2C 04 26 90 00

get response

SCardTransmit (handle 0xEA020000):
transmitted:
80 20 07 00 08 DA E0 BC F7 72 F5 7C 23
received:
90 00

VERIFY 07= IC

SCardTransmit (handle 0xEA020000):
transmitted:
80 20 01 00 08 80 15 7A AF 99 C4 72 5C
received:
90 00

VERIFY 01= ACS1

SCardTransmit (handle 0xEA020000):
transmitted:
80 A4 00 00 02 F0 01
received:
91 01

select file of id = 0xF0 01

SCardTransmit (handle 0xEA020000):
transmitted:
80 B2 00 00 04
received:
04 00 00 00 90 00

read record 0 of selected file

SCardTransmit (handle 0xEA020000):
transmitted:
80 B2 01 00 04
received:
04 00 00 00 90 00

read record 1 of selected file

SCardTransmit (handle 0xEA020000):
transmitted:
80 B2 02 00 04
received:
02 00 00 00 90 00

read record 2 of selected file

SCardTransmit (handle 0xEA020000):
transmitted:
80 A4 00 00 02 F0 01
received:
91 01

select file id=0x F0 01

SCardTransmit (handle 0xEA020000):
transmitted:
80 B2 07 00 04
received:
08 02 00 00 90 00

read record 7 of selected file

SCardTransmit (handle 0xEA020000):
transmitted:
80 A4 00 00 02 F0 02
received:
91 02

SELECT FILE id= 0xF0 02

SCardTransmit (handle 0xEA020000):
transmitted:
80 B2 00 00 F8
received:
47 1E 00 00 56 06 00 00 41 41 00 02 8A 26 19 00 42 42 00 02 8A 26 19 00 42 4C
46 02 8A 26 19 00 43 4B 00 02 8A 26 19 00 43 4C 43 02 8A 26 19 00 43 4C 44 02
8A 26 19 00 43 4E 4C 02 8A 26 19 00 43 52 43 02 8A 26 19 00 46 4C 46 02 8A 26
19 00 46 54 53 02 8A 26 19 00 47 49 47 02 8A 26 19 00 47 53 44 02 8A 26 19 00
4A 4F 42 02 8A 26 19 00 4C 42 00 02 8A 26 19 00 4C 4E 46 02 8A 26 19 00 4D 4D
00 02 8A 26 19 00 4D 4E 52 02 8A 26 19 00 4D 54 4D 02 8A 26 19 00 50 45 4E 02
8A 26 19 00 52 48 44 02 8A 26 19 00 52 4E 52 02 8A 26 19 00 52 4E 57 02 8A 26
19 00 52 52 00 02 8A 26 19 00 53 42 4B 02 8A 26 19 00 53 43 00 02 8A 26 19 00
53 4F 53 02 8A 26 19 00 54 4A 4D 02 8A 26 19 00 55 42 4B 02 8A 26 19 00 56 43
4C 02 8A 26 19 00 57 54 52 02 8A 26 19 00 90 00

read record number 0 of the file

END OF SESSION

2. ANALYSIS

The workflow is very simple. It is based on a series of reading and writing of EF files.

At the beginning of the workflow, a standard mutual authentication is performed to compute session keys K_S. K_S should be dynamic and its freshness is ensure by the card random number generator.

It is hard to say at this stage how the mutual authentication is used but it looks so far that all the communications are be sent in clear so that the session keys are not used. This seems to be used as a basic mutual authentication mechanism.

The rest of the workflow is a sequence of non-ciphered read and updates in a group of 3 files located at the root level of the card:

- ***0xff02****: contains "WTG..." name*
- ***0xf001****: contains several small integer parameters including apparently some kind of size (or date?) on record 7*
- ***0xf002:*** *contains two big chunks of data: one seems to be something containing counters and encrypted data and the other is a counter located in a big memory range (0xF8 bytes)*

At this stage, we need to check how we can realize the mutual authentication.
The mutual authentication in this workflow is simply based on the fact that both side maintains two keysets: K_C and K_T so that the terminals knows the card key in advance and that the card knows the terminal key in advance.

(K_C,K_T) is shared both by card and terminal.

Since we program the card, we apparently do not really care to verify the terminal so that we don't need the terminal keys but we need the card key. Therefore the card key must be located in the terminal memory (e.g. the hard drive), but we will need both K_T and K_C to compute the session keys K_S that the terminal will ultimately verify.

This is enough for cracking authentication.

The rest is pure re-engineering : understanding what are the meaning of the records:

Obviously there must be a date somewhere, a counter and eventually something that is encrypted by the terminal and stored on the card that mix different information (record 00 of file 0xF0 02).

Once both K_C and K_T have been found on the hard drive (or inside the binary game or service WTG application) there should not be any major problems to recreate the card using an applet.

2.1. Mutual Authentication

This is a typical ACOS3 mutual authentication , which is based on standard ISO 7816-8 specifications:

The goal of this authentication is to make sure that the terminal is talking to the right card (and that the card is talking to the right terminal but we do not care about that step)

This authentication system is perfectly void and easily broken if the storage of one of the two parts can be read because the only knowledge of K_C and K_T can allow authentication and both keys are stored on each side.

Since the memory of the terminal may be read, we just need to get the static keys K_C and K_T to make the K_S computation. We note that we can , at card level, skip the rand_C* verification and we do not need to generate a random number.

We will get authenticated by the terminal when the terminal will compare our computation of the session keys K_S with its own computation:

if K_S computed by terminal = K_S computed by card ==> Terminal consider the card as a trusted card

2.2. VERIFY SECRET (CODE)

The terminal also sends a VERIFY command to the card. Of course whatever the terminal sends us through the VERIFY command we will accept it! So this is not a relevant authentication mechanism.

2.3. FILE 0xF0 02 content

This file has a fixed length (0xF8) data.

read	update
47 1E 00 00 55 06 00 00 41 41 00 02	47 1E 00 00 56 06 00 00 41 41 00 02
E1 28 19 00 42 42 00 02	8A 26 19 00 42 42 00 02
E1 28 19 00 42 4C46 02	8A 26 19 00 42 4C 46 02
E1 28 19 00 43 4B 00 02	8A 26 19 00 43 4B 00 02
E1 28 19 00 43 4C 43 02	8A 26 19 00 43 4C 43 02
E1 28 19 00 43 4C 44 02	8A 26 19 00 43 4C 44 02
E1 28 19 00 43 4E 4C 02	8A 26 19 00 43 4E 4C 02
E1 28 19 00 43 52 43 02	8A 26 19 00 43 52 43 02
E1 28 19 00 46 4C 46 02	8A 26 19 00 46 4C 46 02
E1 28 19 00 46 54 53 02	8A 26 19 00 46 54 53 02
E1 28 19 00 47 49 47 02	8A 26 19 00 47 49 47 02
E1 28 19 00 47 53 44 02	8A 26 19 00 47 53 44 02
E1 28 19 00 4A 4F 42 02	8A 26 19 00 4A 4F 42 02
E1 28 19 00 4C 42 00 02	8A 26 19 00 4C 42 00 02
E1 28 19 00 4C 4E 46 02	8A 26 19 00 4C 4E 46 02

E1 28 19 00 4D 4D 00 02 E1 28 19 00 4D 4E 52 02 E1 28 19 00 4D 54 4D 02 E1 28 19 00 50 45 4E 02 E1 28 19 00 52 48 44 02 E1 28 19 00 52 4E 52 02 E1 28 19 00 52 4E 57 02 E1 28 19 00 52 52 00 02 E1 28 19 00 53 42 4B 02 E1 28 19 00 53 43 00 02 E1 28 19 00 53 4F 53 02 E1 28 19 00 54 4A 4D 02 E1 28 19 00 55 42 4B 02 E1 28 19 00 56 43 4C 02 E1 28 19 00 57 54 52 02 E1 28 19 00	8A 26 19 00 4D 4D 00 02 8A 26 19 00 4D 4E 52 02 8A 26 19 00 4D 54 4D 02 8A 26 19 00 50 45 4E 02 8A 26 19 00 52 48 44 02 8A 26 19 00 52 4E 52 02 8A 26 19 00 52 4E 57 02 8A 26 19 00 52 52 00 02 8A 26 19 00 53 42 4B 02 8A 26 19 00 53 43 00 02 8A 26 19 00 53 4F 53 02 8A 26 19 00 54 4A 4D 02 8A 26 19 00 55 42 4B 02 8A 26 19 00 56 43 4C 02 8A 26 19 00 57 54 52 02 8A 26 19 00
read	**update**

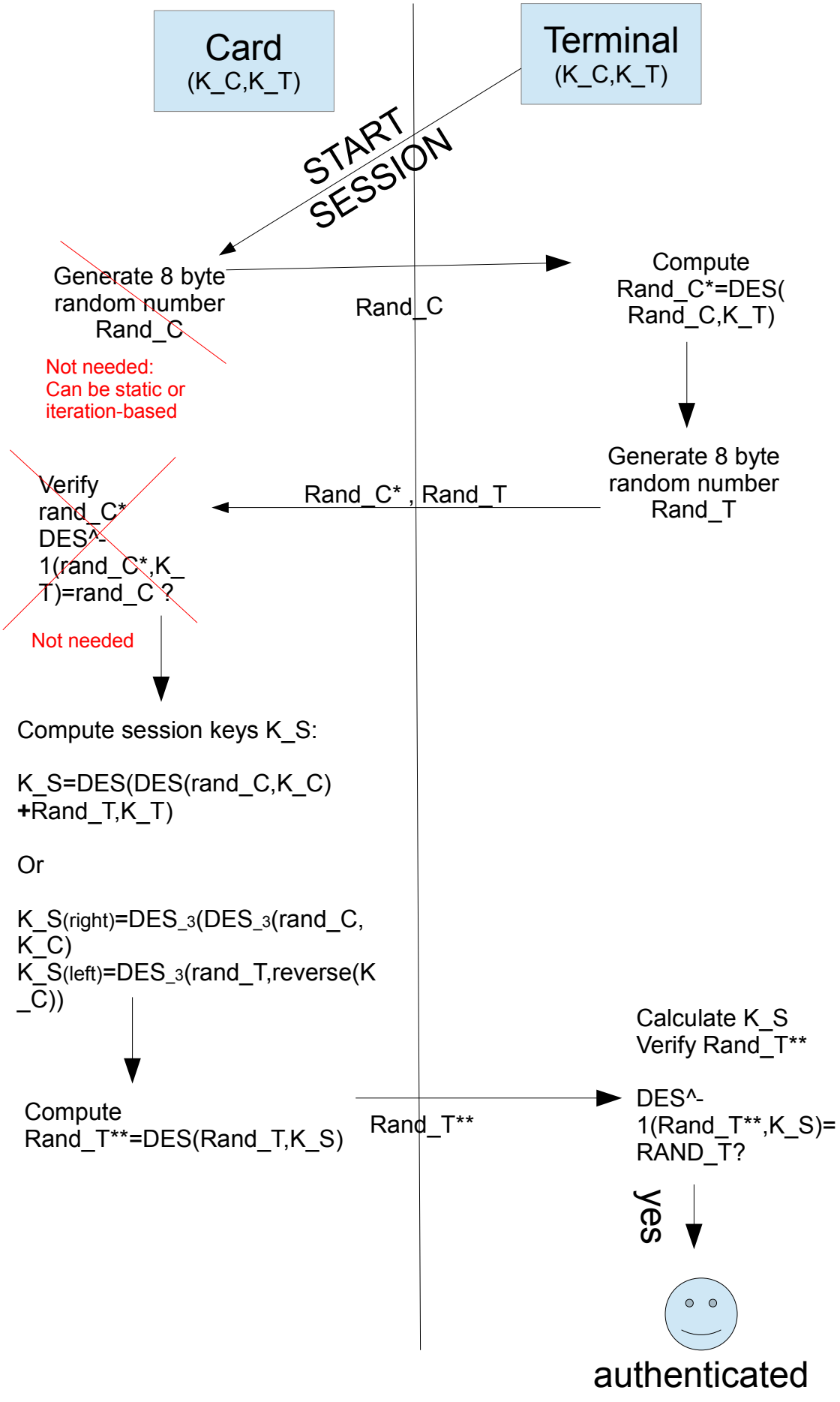
The file seems to maintain a log of counters. This seems to be the records of transactional mechanism.

We also note the presence of a 2 byte header 0xE1 28 which is replace by 0x8A 26 .but the rest of the data are unchanged.

At each read/write of this file, the header is changed and the counter is incremented by the terminal.

This file must be used by the terminal somehow to determine the remaining days: 1 to 60?

Question: how is this file initialized at the beginning?


Card
(K_C,K_T)
Terminal
(K_C,K_T)
START SESSION
Generate 8 byte random number Rand_C
Not needed: Can be static or iteration-based
Rand_C
Compute Rand_C*=DES(Rand_C,K_T)
Generate 8 byte random number Rand_T
Rand_C* , Rand_T
Verify rand_C* DES^-1(rand_C*,K_T)=rand_C ?
Not needed
Compute session keys K_S:
K_S=DES(DES(rand_C,K_C) +Rand_T,K_T)
Or
K_S(right)=DES_3(DES_3(rand_C, K_C)
K_S(left)=DES_3(rand_T,reverse(K_C))
Compute Rand_T**=DES(Rand_T,K_S)
Rand_T**
Calculate K_S
Verify Rand_T**
DES^-1(Rand_T**,K_S)= RAND_T?
yes
authenticated