```
 1 This is the protocol of an authentification using
   key 1 (read & write access) that was
 2 a CHANGED DES key (key value 'd10023456789abcd').
   After the 'setKeyVersion to 0' the resulting key is
    'd00022446688aacc'.
 3
 4 method: authenticateD40
 5 key length: 8 data: d10023456789abcd keyNo: 1
 6 method: authenticateD40
 7 step 01 get encrypted rndB from card
 8 method: authenticateD40
 9 - send auth apdu   apdu     length: 7 data:
   900a0000010100
10 method: authenticateD40
11 - receive response response length: 10 data:
   eb0533b4bc89afcf91af
12 method: authenticateD40
13 step 02 get the encrypted rndB from response data
14 method: getData
15 responseAPDU length: 10 data: eb0533b4bc89afcf91af
16 method: getData
17 responseData length: 8 data: eb0533b4bc89afcf
18 method: authenticateD40
19 - encryptedRndB length: 8 data: eb0533b4bc89afcf
20 method: authenticateD40
21 step 03 setKeyVersion to 00 for DES keys
22 method: authenticateD40
23 - DES key provided       length: 8 data:
   d10023456789abcd
24 method: setKeyVersion
25 a length: 8 data: d10023456789abcd offset: 0 length
   : 8 version: 0
26 method: authenticateD40
27 - DES key w/keyVersion 0 length: 8 data:
   d00022446688aacc
28 method: authenticateD40
29 step 04 get a TDES key from the DES key
30 method: getTDesKeyFromDesKey
31 key length: 8 data: d00022446688aacc
32 method: getTDesKeyFromDesKey
33 TDES key length: 24 data:
```

```
33 d00022446688aaccd00022446688aaccd00022446688aacc
34 method: authenticateD40
35 - DES key   length: 8 data: d00022446688aacc
36 method: authenticateD40
37 - TDES key length: 24 data:
   d00022446688aaccd00022446688aaccd00022446688aacc
38 method: authenticateD40
39 step 06 decrypt the encRndB using TripeDES.decrypt
   with key key length: 8 data: d00022446688aacc iv0
   length: 8 data: 0000000000000000
40 method: authenticateD40
41 - encrypted rndB length: 8 data: eb0533b4bc89afcf
42 method: authenticateD40
43 - decrypted rndB length: 8 data: c4a3468fb0d87e74
44 method: authenticateD40
45 step 06 rotate the decrypted rndB by 1 position/
   byte to the left
46 method: authenticateD40
47 - rndB              length: 8 data:
   c4a3468fb0d87e74
48 method: rotateLeft
49 data length: 8 data: c4a3468fb0d87e74
50 method: authenticateD40
51 - rndB left rotated length: 8 data:
   a3468fb0d87e74c4
52 method: authenticateD40
53 step 07 generate a random rndA
54 method: getRandomData
55 key length: 8 data: 0000000000000000
56 method: getRandomData
57 length: 8
58 method: authenticateD40
59 - rndA length: 8 data: 45cc39928713e1c0
60 method: authenticateD40
61 step 08 concatenate rndA || rndB left rotated
62 method: authenticateD40
63 - rndA || rndB left rotated length: 16 data:
   45cc39928713e1c0a3468fb0d87e74c4
64 method: authenticateD40
65 step 09 copy encryptedRndB to iv1 from position 0
   to 8
```

```
66 method: authenticateD40
67 iv1 length: 8 data: eb0533b4bc89afcf
68 method: authenticateD40
69 step 09 copy encryptedRndB to iv1 from position
70 method: authenticateD40
71 step 10 encrypt rndA || rndB left rotated
72 method: authenticateD40
73         Note: we are encrypting the data by
   DEcrypting the plaintext due to PICC
   characteristics
74 method: authenticateD40
75 using mode case SEND_MODE = XOR w/ previous
   ciphered block --> decrypt
76 method: authenticateD40
77 step 10 encryption magic starting
    **********************
78 method: tripleDesSendModeDecryption
79 *** start of the manual decryption ***
80 method: tripleDesSendModeDecryption
81 the ciphertext is 16 bytes long so we need to run
   2 rounds to decrypt (length / 8)
82 method: tripleDesSendModeDecryption
83 ******** manual decryption text start ********
84 method: tripleDesSendModeDecryption
85 SEND mode means: XORing the ciphertext with
   previous ciphered block, than DEcrypt
86 method: tripleDesSendModeDecryption
87 tdesKey length: 24 data:
   d00022446688aaccd00022446688aaccd00022446688aacc
88 method: tripleDesSendModeDecryption
89 1 starting with an empty 'cipheredBlock' of 8
   bytes length = DES block length
90 method: tripleDesSendModeDecryption
91 cipheredBlock    length: 8 data: 0000000000000000
92 method: tripleDesSendModeDecryption
93 2 split the ciphertext into blocks of 8 bytes
94 method: tripleDesSendModeDecryption
95 ciphertext      length: 16 data:
   45cc39928713e1c0a3468fb0d87e74c4
96 method: tripleDesSendModeDecryption
97 ciphertextBlock1 length: 8 data: 45cc39928713e1c0
```

```
 98 method: tripleDesSendModeDecryption
 99 ciphertextBlock2 length: 8 data: a3468fb0d87e74c4
100 method: tripleDesSendModeDecryption
101 3 XORing ct1 with cipheredBlock
102 method: tripleDesSendModeDecryption
103 ct1 Xored        length: 8 data: 45cc39928713e1c0
104 method: tripleDesSendModeDecryption
105 4 decrypt ct1Xored using TripleDES.decrypt
106 method: tripleDesSendModeDecryption
107 ct1Xored decrypt length: 8 data: 88e199b02da83367
108 method: tripleDesSendModeDecryption
109 5 copy ct1XoredDecrypted to cipheredBlock
110 method: tripleDesSendModeDecryption
111 cipheredBlock    length: 8 data: 88e199b02da83367
112 method: tripleDesSendModeDecryption
113 6 XORing ct2 with cipheredBlock
114 method: tripleDesSendModeDecryption
115 ct2Xored         length: 8 data: 2ba71600f5d647a3
116 method: tripleDesSendModeDecryption
117 7 decrypt ct2Xored using TripleDES.decrypt
118 method: tripleDesSendModeDecryption
119 ct2 Xored decrypt length: 8 data: 557208d962ae4b4f
120 method: tripleDesSendModeDecryption
121 8 Note: for more data this would be extended but
    we are ready now
122 method: tripleDesSendModeDecryption
123 9 concatenate ct1XoredDecrypted and
    ct2XoredDecrypted to plaintext
124 method: tripleDesSendModeDecryption
125 plaintext length: 16 data:
    88e199b02da83367557208d962ae4b4f
126 method: tripleDesSendModeDecryption
127 ******** manual decryption text end **********
128 method: authenticateD40
129 XOR w/ previous ciphered block --> decrypt
130 method: authenticateD40
131 data before XORing data length: 16 data:
    45cc39928713e1c0a3468fb0d87e74c4 cipheredBlock
    length: 8 data: 0000000000000000
132 method: authenticateD40
133 running a 2 round loop to XOR rndArndBLeftRotated
```

```
133 with the previous cipheredBlock and DEcrypt the
    block using TripleDES
134 method: authenticateD40
135 The outer loop is running for i=0 to <16 in steps
    of 8
136 method: authenticateD40
137 outer loop i: 0
138 method: authenticateD40
139 The inner loop is running for j=0 to <8 in steps
    of 1
140 method: authenticateD40
141 TripleDES.decrypt cipheredBlock length: 8 data:
    88e199b02da83367
142 method: authenticateD40
143  copying cipheredBlock to ciphertext from i = 0
    length 8
144 method: decrypt
145  ciphertext length: 16 data:
    88e199b02da8336700000000000000000
146 method: authenticateD40
147 outer loop i: 8
148 method: authenticateD40
149 The inner loop is running for j=0 to <8 in steps
    of 1
150 method: authenticateD40
151 TripleDES.decrypt cipheredBlock length: 8 data:
    557208d962ae4b4f
152 method: authenticateD40
153  copying cipheredBlock to ciphertext from i = 8
    length 8
154 method: decrypt
155  ciphertext length: 16 data:
    88e199b02da83367557208d962ae4b4f
156 method: authenticateD40
157 step 10 encryption magic ending
       ********************
158 method: authenticateD40
159 manual decryption: SUCCESS
160 method: authenticateD40
161 - encrypted rndA || rndB left rotated length: 16
    data: 88e199b02da83367557208d962ae4b4f
```

```
162 method: authenticateD40
163 step 11 send the encrypted data to the PICC using
    the 0xAF command (more data)
164 method: authenticateD40
165 - send auth apdu   apdu     length: 22 data:
    90af00001088e199b02da83367557208d962ae4b4f00
166 method: authenticateD40
167 - receive response response length: 10 data:
    6ccc27d21352c5ee9100
168 method: authenticateD40
169 step 12 the response data is the encrypted rndA
    from the PICC
170 method: authenticateD40
171         Note: the received (encrypted) rndA is
    left rotated
172 method: getData
173 responseAPDU length: 10 data: 6ccc27d21352c5ee9100
174 method: getData
175 responseData length: 8 data: 6ccc27d21352c5ee
176 method: authenticateD40
177 - encrypted rndA left rotated length: 8 data:
    6ccc27d21352c5ee
178 method: authenticateD40
179 encryptedRndA length: 8 data: 6ccc27d21352c5ee
180 method: authenticateD40
181 The iv is set to 8 * 0x00
182 method: authenticateD40
183 iv0 length: 8 data: 0000000000000000
184 method: authenticateD40
185 step 13 decrypt the encrypted rndA left rotated
    using TripeDES.decrypt with key key length: 8 data
    : d00022446688aacc iv0 length: 8 data:
    0000000000000000
186 method: authenticateD40
187 - encrypted left rotated rndA length: 8 data:
    eb0533b4bc89afcf
188 method: authenticateD40
189 - decrypted left rotated rndA length: 8 data:
    cc39928713e1c045
190 method: authenticateD40
191 step 14 rotate decrypted left rotated rndA to
```

```
191 RIGHT
192 method: authenticateD40
193 - decrypted rndA length: 8 data: 45cc39928713e1c0
194 method: authenticateD40
195 step 15 compare self generated rndA with rndA
    received from PICC
196 method: authenticateD40
197 - rndA generated length: 8 data: 45cc39928713e1c0
198 method: authenticateD40
199 - rndA received  length: 8 data: 45cc39928713e1c0
200 method: authenticateD40
201 - rndA generated and received are equals: true
202 method: authenticateD40
203 step 16 generate the DES Session key from rndA and
     rndB
204 method: authenticateD40
205 - rndA           length: 8 data: 45cc39928713e1c0
206 method: authenticateD40
207 - rndB           length: 8 data: c4a3468fb0d87e74
208 method: getSessionKey
209 rndA length: 8 data: 45cc39928713e1c0 rndB length
    : 8 data: c4a3468fb0d87e74
210 method: authenticateD40
211 - This are the first 4 bytes of rndA and rndB, the
     DES Session key is
212 method: authenticateD40
213 - rndA first 4 bytes || rndB first 4 bytes
214 method: authenticateD40
215 - rndA first 4 Bytes    45CC3992
216 method: authenticateD40
217 - rndB first 4 Bytes            C4A3468F
218 method: authenticateD40
219 - SessionKey is 8 Bytes 45CC3992C4A3468F (length:
    8)
220 method: authenticateD40
221 **** auth result ****
222 method: authenticateD40
223 *** AUTHENTICATED ***
224 method: authenticateD40
225 **********************
226
```