

Introduction to RACS

Remote APDU Call Secure

Introducing RACS

- The RACS protocol has been designed for efficient and secure remote use of secure elements via the internet.
- It also provides smartcard reader virtualization, and therefore facilitates secure elements deployment in environments dealing with virtual machines and cloud computing.

ISO7816 APDU
TLS
TCP
IP

RACS Uniform Resource Identifier

- RACS client establishes a secure (TLS) connection with a remote server, it collects the list of hosted secure elements, and thereafter it powers a secure element, resets the device, and exchanges APDU requests and responses.
- RACS uses an URI (*Uniform Resource Identifier*), such as :
 - **ServerName:Port/SEID**
- The URI comprises:
 - the server name or IP address,
 - the TCP port
 - and a SE identifier (the SEID).
- Therefore it creates a concept somewhat similar to a WEB of secure elements, or a WEB of cryptographic procedures.

A PKI Infrastructure

- In order to enforce strong mutual authentication both RACS client and server are equipped with X509 certificates, dealing with asymmetric cryptography (RSA or Elliptic Curves).
- The client SUBJECT attribute, more precisely the *Common Name* (CN) field of this attribute, identifies a legitimate client, and is associated within the server to an index UID, the User Identifier.

How to allocate SEID ?

Reader Serial Number: some readers have S/N that can be read via PC/SC

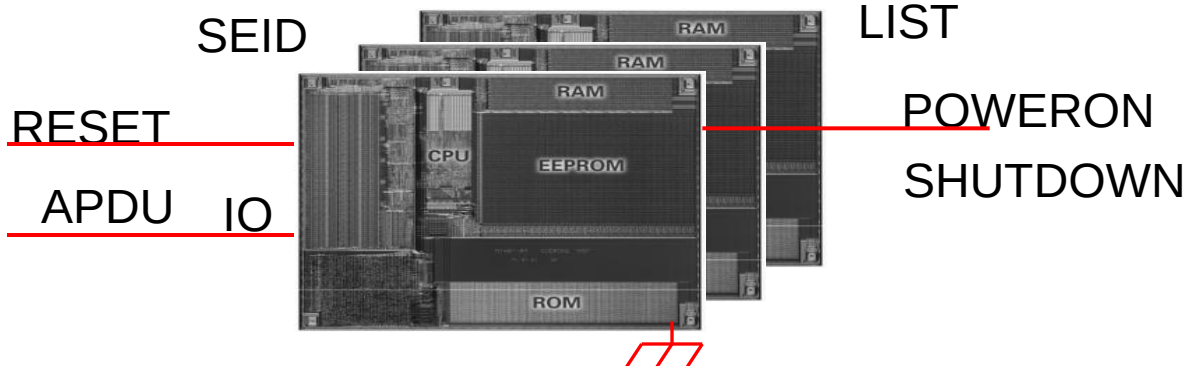


Smartcard Reader Index: smartcard readers are identified by an index starting from 0

RACS Requests and Responses

```
BEGIN MyLabel
GET-VERSION APPEND
LIST APPEND
POWERON MySEID APPEND
RESET MySEID
APDU MySEID
00A40400074A544553543030
END
```

```
BEGIN MyLabel
+002 001 0.2
+004 002 MySEID OtherSEID
+008 003 MySEID is powered on
+005 004 MySEID reseted
+006 005 9000
END
```



APDU Command

```
1. BODY = empty;
2. SW    = empty;
3. DoIt = true;
3. DO
4. { iso7816-response = send(iso7816-request);
5.   body || sw1 || sw2 = iso7816-response;
6.   IF ( (first request) &&
          (iso7816-request.size==5) &&
          (body==empty) && (sw1==6C) )
8.     { iso7816-request.P3 = sw2 ; }
6.   ELSE
7.     { SW = sw1 || sw2
8.       BODY = BODY || body;
9.       IF (sw1 == MORE)
10.        { iso7816-request = FETCH || sw2 ; }
11.       ELSE
12.        { DoIt=false;}
13.     }
14. }
15. While (DoIt == true)

16. iso7816-response = BODY || SW ;
17. IF (SW != CONTINUE) Error ;
18. ELSE      No Error;
```

FETCH
00 C0 00 00

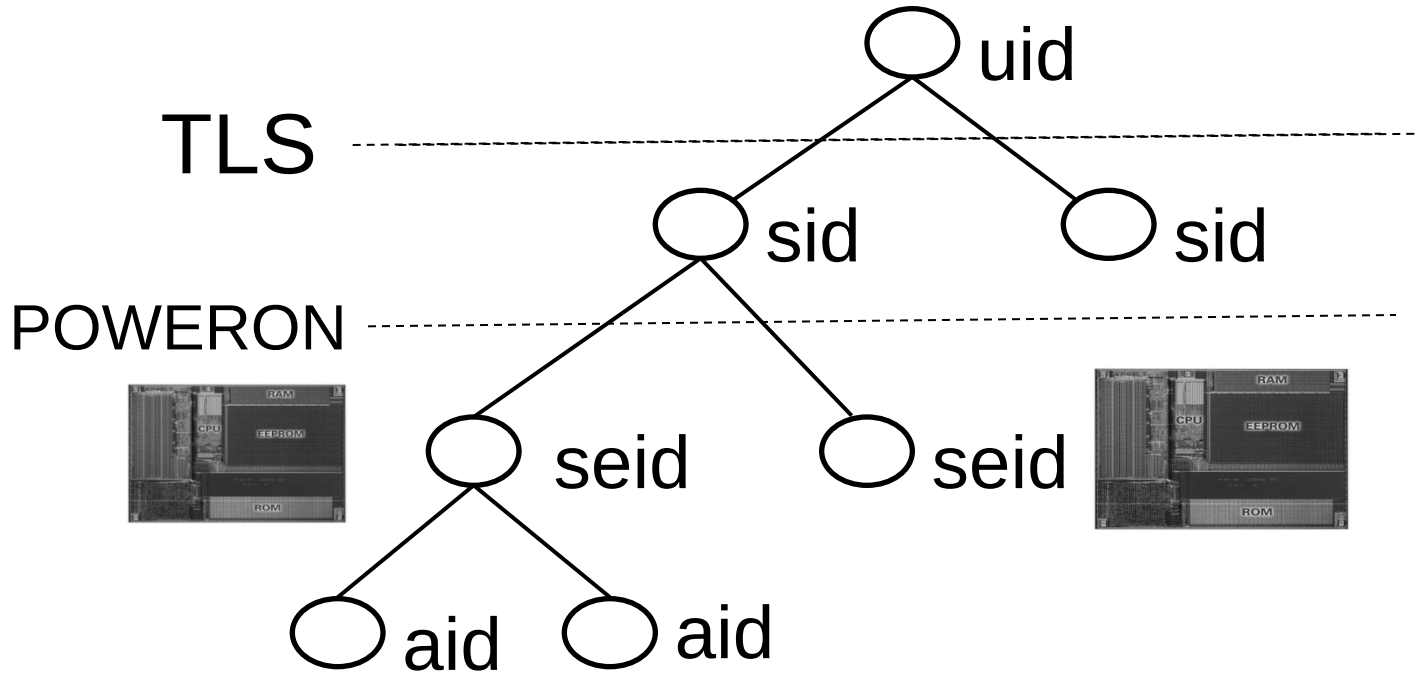
MORE
61

CONTINUE
9000

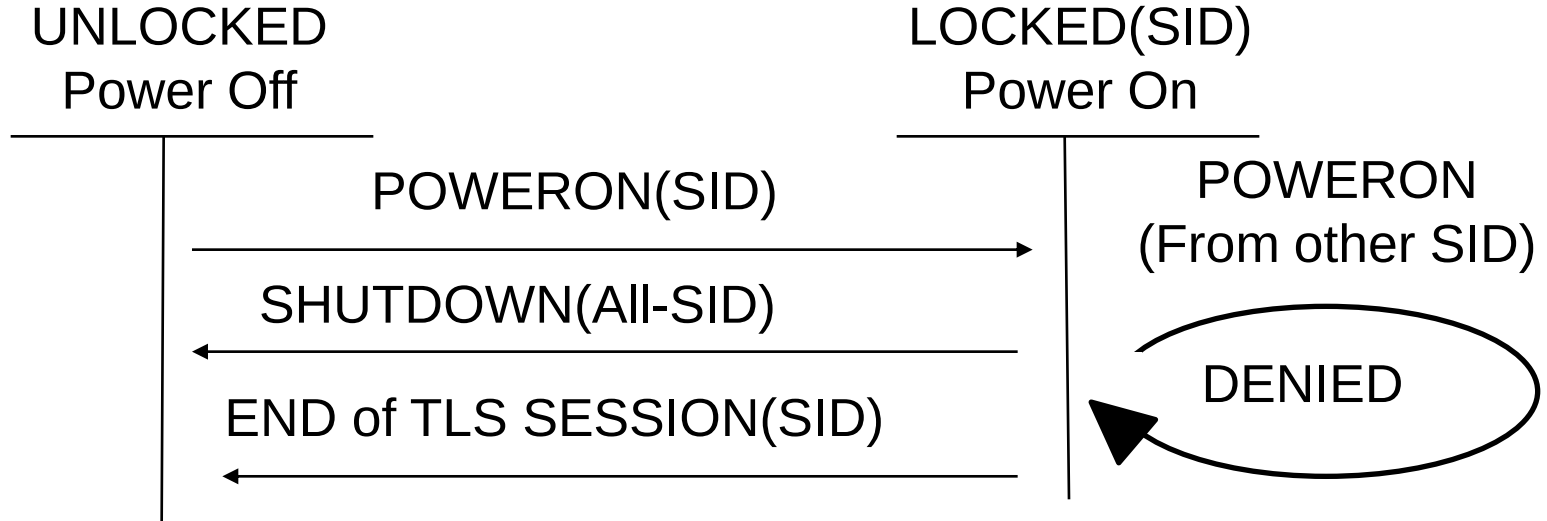
Security Policy

- Only users equipped with valid certificates successfully establish TLS sessions.
- A user identifier (UID) is derived from the certificate Common Name (CN) attribute.
- Each TLS session is identified by a unique identifier (the SID).
- Every secure element has two states, UNLOCKED and LOCKED.
 - The SHUTDOWN command forces the UNLOCKED state; the POWERON command switches the SE state from LOCKED to UNLOCKED.
 - In the locked state the SE may only be used by the SID that previously locked it.
- At the end of a TLS session, all used SEs are unpowered and unlocked.

UID, SID, AID



SEID Locking



Access Control

- The server manages two kinds of table:
 - The *Users-Table* stores for each CN a list of authorized SEIDs.
 - Each SEID is associated to a *SEID-Table*, storing for every AID (Application Identifier of embedded application) a list of authorized CNs.

SEID files

```
"SCR3310 Reader"  
"CardMan 5x21-CL"
```

type.txt file
List of reader name with SN

```
21120548219311 obelix  
21120837203028 paycard
```

readersn.txt file
List of tuples (SN,SEID)

```
//WENEO                                ATR                                Issuer Security Domain (ISD)  
3BF91800008031FE4580574E454F574156457D  A0000000003000000  
//GX40  
3B7D960|00080318065B08311C0C883009000    A0000000018434D00  
atr.txt File (DEPRECATED)
```

TypeTable (type.txt)

- According to the PCSC standard each smartcard reader MAY manage three attributes
 - Vendor
 - Type
 - Serial Number
- Hypothesis: The Type attribute is unique for a class of reader
- The TypeTable indicates that a given type is associated to a true unique Reader Serial Number (SN)
 - Some readers don't support the SN option or has a fix (fake) SN
- Example
 - "Weneo Reader"
 - "SCR33x USB Smart Card Reader"
 - "SDI010 Contactless Reader"
 - "SDI010 Smart Card Reader"

ReaderSN table

- Establishes the relationship between reader serial number and SEID
- For some dual interfaces readers (contact / contactless) a single SN may be shared by two different SEIDs
- Example
 - // lecteur scr133
 - 21120548219311 obelix
 - // Lecteur OMNIKEY 5321
 - OKCM002040110172830810179479861 paycard
 - // LECTEUR SID010
 - 21120837203028 paycard "SDI010 Contactless Reader"
 - 21120837203028 contact "SDI010 Smart Card Reader "

AtrTable (atr.txt)

- Hypothesis: Each SE class is uniquely identified by an ATR
 - See for <https://smartcard-atr.apdu.fr/>
- Each SE has a Issuer Security Domain (ISD), associated to an Application Identifier (AID) 16 bytes at the most
- The AtrTable manages the relationship between ATR and ISD
- Example
 - // WENEO
 - 3BF91800008031FE4580574E454F574156457D A000000003000000
 - // GX40
 - 3B7D96000080318065B08311C0C883009000 A000000018434D00
 - // JCOP41 NFC
 - 3BFA1800008131FE454A434F503431563232319D A000000003000000
- **This table is deprecated**

Access Control Files

```
bob      2 obelix  paycard
alice    2 asterix muscle
admin    4 obelix asterix muscle paycard
```

users.txt file

```
A0000000003000000  no 1 admin
A000000000101      no 2 bob alice:filter.txt
default            no 1 admin
```

seid.txt file

```
// The filter.txt file
// Mask          APDU-Prefix
FFFF0000         A0200000
00FF0000         00D80000
```

Filter.txt file

UserTable (users.txt)

- The Common Name (CN) attribute is found in the client X509 certificate
- Each CN is associated to an UserID, an integer value (from 0 to n)
- The UserID (UID) is allocated from the first CN occurrence in the UserTable.
- A CN not referenced in this table has no UID, and therefore has no access to SE
- Example:
 - client 4 725 obelix asterix paycard // 4 SE, 725 obelix asterix paycard
 - client2 0 // 0 SE
 - Admin all // All SE

SEID Table

- Each AID is associated to a white or black access list
- A default option is available
- AID no listed in the SEID-Table are accessed according to the ***se_default_access*** attribute defined in the config.txt file (yes=access, no=access denied)
- Example:
 - // AID or default, no=WhiteList yes=BlackList , #CN, list of CN
 - A0000000003000000 no 3 pascal client admin
 - A000000000101 no 3 pascal client admin
 - default no 2 pascal admin

CardSn table

- Establishes the relationship between secure element server name (SN) and SEID
- The Server Name is defined by ATR historical bytes.
- Example
 - //CARD_SN SEID
 - key1.com key1

LIST of Commands

COMMAND	SEID	
GET-VERSION	no	Return current version
SET-VERSION	no	Set version
LIST	no	List authorized SEID
RESET	yes	Reset secure element
APDU	yes	Send APDU to secure element
SHUTDOWN	yes	Shutdown secure element
POWERON	yes	Power-on secure element
ECHO	no	Do Echo
SEN	yes	Associates (SEN,AID) to SEID
GET-SEN	yes	Return SEN associated to SEID

More Details

[https://datatracker.ietf.org/doc/html
/draft-urien-core-racs-19.txt](https://datatracker.ietf.org/doc/html/draft-urien-core-racs-19.txt)