**Application Selection**

Application Selection is the first step after the Answer to Reset. It has the function to select the ADF for the transaction process.  
  
There are two ways to get the right ADF. If the card supports the Payment System Environment (PSE), the terminal reads out the necessary information to select the ADF.  
If there is no PSE, the terminal will use a list of AIDs and get the right by trying.

**Using the PSE method**

/\*\*

\* Send SELECT APDU

\*/

EMV.prototype.select = function(dfname, first) {

var fci = this.card.[sendApdu](http://www.openscdp.org/scsh3/card.html#sendApdu)

(0x00, 0xA4, 0x04, (first ? 0x00 : 0x02), dfname, 0x00);

return(fci);

}

/\*\*

\* Send READ RECORD APDU

\*

\* Return empty ByteString if no data was read

\*/

EMV.prototype.readRecord = function(sfi, recno) {

var data = this.card.sendApdu(0x00, 0xB2, recno, (sfi << 3) | 0x04, 0);

if (this.card.SW1 == 0x6C) {

var data = this.card.sendApdu

(0x00, 0xB2, recno, (sfi << 3) | 0x04, this.card.SW2);

}

return(data);

}

First we select the PSE (1PAY.SYS.DDF01)

/\*\*

\* Select and read Payment System Environment on either

\* contact or contactless card

\*/

EMV.prototype.selectPSE = function(contactless) {

this.PSE = null;

var dfname = (contactless ? EMV.PSE2 : EMV.PSE1);

var fci = this.select(dfname, true);

print(fci);

if (fci.length == 0) {

GPSystem.trace("No " + dfname.toString(ASCII) + " found");

return;

}

A2 C: 00 A4 04 00 - SELECT Lc=14

0005 31 50 41 59 2E 53 59 53 2E 44 44 46 30 31 1PAY.SYS.DDF01

Le=0

R: SW1/SW2=9000 (Normal processing: No error) Lr=23

0000 6F 15 84 0E 31 50 41 59 2E 53 59 53 2E 44 44 46 o...1PAY.SYS.DDF

0010 30 31 A5 03 88 01 01 01.....

The selection was successful (SW1/SW2=9000), so the card supports the PSE method.

**Get the SFI**

To process we need the SFI.  
Therefore we have to decode first the returned FCI.

// Decode FCI Template

var tl = new [TLVList](http://www.openscdp.org/scsh3/tlvlist.html)(fci, TLV.EMV);

var t = tl.[index](http://www.openscdp.org/scsh3/tlvlist.html#index)(0);

var t:

6F 15 84 0E 31 50 41 59 2E 53 59 53 2E 44 44 46 30 31 A5 03 88 01 01

assert(t.getTag() == EMV.FCI); // EMV.FCI = 0x6F

var tl = new TLVList(t.getValue(), TLV.EMV);

assert(tl.length >= 2);

var tl:

84 0E 31 50 41 59 2E 53 59 53 2E 44 44 46 30 31 A5 03 88 01 01

// Decode DF Name

t = tl.index(0);

var t:

84 0E 31 50 41 59 2E 53 59 53 2E 44 44 46 30 31

assert(t.getTag() == EMV.DFNAME); // EMV.DFNAME = 0x84;

// Decode FCI Proprietary Template

t = tl.index(1);

var t:

A5 03 88 01 01

assert(t.getTag() == EMV.FCI\_ISSUER); // EMV.FCI\_ISSUER = 0xA5;

var tl = new TLVList(t.getValue(), TLV.EMV);

// Decode SFI of the Directory Elementary File

t = tl.index(0);

var t:

88 01 01

assert(t.getTag() == EMV.SFI); // EMV.SFI = 0x88;

var sfi = t.getValue();

assert(sfi.length == 1);

sfi = sfi.byteAt(0);

sfi: 0x01

this.PSE = new Array();

**Read Records**

Now we create a loop to read all records in this AEF, started with record number 1.

// Read all records from Directory Elementary File

var recno = 1;

do {

var data = this.readRecord(sfi, recno++);

if (data.length > 0) {

var tl = new TLVList(data, TLV.EMV);

assert(tl.length == 1);

var t = tl.index(0);

assert(t.getTag() == EMV.TEMPLATE);

var tl = new TLVList(t.getValue(), TLV.EMV);

assert(tl.length >= 1);

for (var i = 0; i < tl.length; i++) {

var t = tl.index(i);

assert(t.getTag() == 0x61);

this.PSE.push(new TLVList(t.getValue(), TLV.EMV));

}

}

} while (data.length > 0);

}

At this time we have two entries in this PSE:

61 1F 4F 08 A0 00 00 00 25 01 05 01 50 10 50 65 72 73 6F 6E 61 6C 20 41 63 63 6F 75 6E

74 87 01 01

and

61 1E 4F 07 A0 00 00 00 29 10 10 50 10 50 65 72 73 6F 6E 61 6C 20 41 63 63 6F 75 6E 74

87 01 02

**Check Priority**

Every entry contains an AID. Now we proof which of them have they highest priority.

/\*\*

\* Return array of PSE entries or null if none defined

\*/

EMV.prototype.getPSE = function() {

return this.PSE;

}

/\*\*

\* Return AID of application with highest priority or null if no PSE defined

\*/

EMV.prototype.getAID = function() {

var prio = 0xFFFF;

var aid = null;

var pse = e.getPSE();

if (pse == null) {

return null;

}

// Iterate through PSE entries

for (var i = 0; i < pse.length; i++) {

var t = pse[i].find(EMV.AID);

assert(t != null);

var entryAid = t.getValue();

print(entryAid);

var t = pse[i].find(EMV.LABEL);

assert(t != null);

print(t.getValue().toString(ASCII));

var entryPrio = 0xFFFE;

var t = pse[i].find(EMV.PRIORITY);

if (t != null) {

entryPrio = t.getValue().toUnsigned();

entryPrio &= 0x0F;

}

if (entryPrio < prio) {

prio = entryPrio;

aid = entryAid;

}

}

this.cardDE[EMV.AID] = aid;

return aid;

}

In the first run the value for the entryPrio is: 0x01. 0x01 &= 0x0F is 0x01. 0x01 is less than 0xFFFF and from this it follows that 0x01 is now the prio and the aid is now the entryAid.  
In the second run the value for the entryPrio is: 0x02. 0x02 &= 0x0F is 0x02. 0x02 is greater than 0x01, the priority of this entry is lower.

**Final Selection**

Select The ADF with the given AID.

/\*\*

\* Select application and return FCI

\*/

EMV.prototype.selectADF = function(aid) {

var fci = this.select(aid, true);

print(fci);

this.cardDE[EMV.AID] = aid;

}

**Using a list of AIDs**

First we select the PSE (1PAY.SYS.DDF01)

/\*\*

\* Select and read Payment System Environment on either

\* contact or contactless card

\*/

EMV.prototype.selectPSE = function(contactless) {

this.PSE = null;

var dfname = (contactless ? EMV.PSE2 : EMV.PSE1);

var fci = this.select(dfname, true);

print(fci);

if (fci.length == 0) {

GPSystem.trace("No " + dfname.toString(ASCII) + " found");

return;

}

96 C: 00 A4 04 00 - SELECT Lc=14

0005 31 50 41 59 2E 53 59 53 2E 44 44 46 30 31 1PAY.SYS.DDF01

Le=0

R: SW1/SW2=6A82 (Checking error: File not found) Lr=0

No 1PAY.SYS.DDF01 found

The selection failed because the ADF doesn't exist (SW1/SW2=6A82). The terminal uses now a list of AIDs to select the right ADF.

The terminal starts with the first AID in the Array. If the selection failed the terminal tries the next AID.

/\*\*

\* Try a list of predefined AID in order to select an application

\*/

EMV.prototype.tryAID = function() {

for (var i = 0; i < EMV.AIDLIST.length; i++) {

var le = EMV.AIDLIST[i];

var aid = new ByteString(le.aid, HEX);

var fci = this.select(aid, true);

if (fci.length > 0) {

this.cardDE[EMV.AID] = aid;

return;

}

}

}

// EMV.AIDLIST:

EMV.AIDLIST = new Array();

EMV.AIDLIST[0] = { aid : "A00000002501", partial : true, name : "AMEX" };

EMV.AIDLIST[1] = { aid : "A0000000031010", partial : false, name : "VISA" };

EMV.AIDLIST[2] = { aid : "A0000000041010", partial : false, name : "MC" };

96 C: 00 A4 04 00 - SELECT Lc=6

0005 A0 00 00 00 25 01 ....%.

Le=0

R: SW1/SW2=6A82 (Checking error: File not found) Lr=0

96 C: 00 A4 04 00 - SELECT Lc=7

0005 A0 00 00 00 03 10 10 .......

Le=0

R: SW1/SW2=9000 (Normal processing: No error) Lr=42

0000 6F 28 84 07 A0 00 00 00 03 10 10 A5 1D 50 04 56 o(...........P.V

0010 49 53 41 87 01 01 9F 38 0C 9F 33 03 9F 1A 02 9F ISA....8..3.....

0020 35 01 9F 40 05 5F 2D 02 64 65 5..@.\_-.de