**Read Application Data**

This function reads all data from the Application File Locator (AFL). The AFL identifies the files and records which are necessary for the transaction.

**AFL**

**Structure**

Every 4 bytes stick together.

|  |
| --- |
|  |
| **Byte** | **Function** |
| 1 | Short File Identifier The five most significant bits are the SFI. The three least significant bits are set to zero. Example:  '10'  00010000 >>3  = 00000010  = '02' = SFI |
| 2 | Start Record |
| 3 | End Record |
| 4 | Number of records included in data authentication beginning from the Start Record |

**Script**

/\*\*

\* Read application data as indicated in the Application File Locator

\* Collect input to data authentication

\*

\*/

EMV.prototype.readApplData = function() {

// Application File Locator must exist

assert(typeof(this.cardDE[EMV.AFL]) != "undefined");

var afl = this.cardDE[EMV.AFL];

// Must be a multiple of 4

assert((afl.length & 0x03) == 0);

// Collect input to data authentication

var da = new [ByteBuffer](http://www.openscdp.org/scsh3/bytebuffer.html)();

while(afl.length > 0) {

var sfi = afl.[byteAt](http://www.openscdp.org/scsh3/bytestring.html#byteAt)(0) >> 3; // Short file identifier

var srec = afl.byteAt(1); // Start record

var erec = afl.byteAt(2); // End record

var dar = afl.byteAt(3); // Number of records included in

// data authentication

for (; srec <= erec; srec++) {

// Read all indicated records

var data = this.readRecord(sfi, srec);

print(data);

// Decode template

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

assert(tl.length == 1);

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

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

// Add data authentication input

if (dar > 0) {

if (sfi <= 10) { // Only value

da.[append](http://www.openscdp.org/scsh3/bytebuffer.html#append)(t.getValue());

} else { // Full template

da.append(data);

}

dar--;

}

// Add card based data elements to internal list

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

this.addCardDEFromList(tl);

}

// Continue with next entry in AFL

afl = afl.[bytes](http://www.openscdp.org/scsh3/bytestring.html#bytes)(4);

}

this.daInput = da.toByteString();

print(this.daInput);

}

The function always extract the first 4 bytes of the bytestring. That is why the length of the AFL bytestring has to be a multiple of 4. With a read record command and the bytes from the start record to the end record will be read. If start and end record are equal, there is only one record. The records will be stored in the cardDE array without the EMV.TEMPLATE (tag: '70' and length).  
The data marked by the fourth byte will be stored in the da bytebuffer. If the SFI is greater as 10, the whole record will be stored in the bytebuffer. The data in the bytebuffer will be need for dynamic data authentication.  
After this the 4 bytes of the AFL bytestring will be removed. This process will be repeated untill the AFL bytestring is empty.