PKI MRTD API Documentation

PKI provides a Machine Readable Travel Document interface, available through the REST protocol.
This interface generates ICAO 9303 datagroups and SOD, to be inserted on a smartcard to provide user identification services.
The generated data is fully compatible with JMRTD (http://jmrtd.org) and probably with other MRTD implementations.

All parameters must be passed as UTF-8 (URL encoded depending on the HTTP GET/POST method used).

Requesting an MRTD blob

To request an MRTD blob, following HTTP request should be performed:
wget --user=admin --password=<pki_admin_password> "https://<pki_host>/pki/?action=mrtd&cn=Test&serial=1234567890ABC&doe=010101&direct=1"

Note that this is explained using wget and GET HTTP parameters for the sake of simplicity, but HTTP POST requests can be performed instead, with the following request parameters:

- cn: requested CN (automatically translitterated to ASCII and translated as surname/ givenname by the system)
- serial: user's personal number (employeeNumber, national ID number, social security number, ...)
- doe: date of expiry of the digital document, in YYMMDD format (010101 means January, 1st, 2001)
- direct: 1 (to activate REST mode)
- action: mrtd (to request an MRTD)

This request returns an application/json reply, containing a JSON message with either "OK" and a Base64 encoded pkcs12, or an error message in case of error.

Note
To be MRTD compliant, the reply should be stored as following files on card (in binary mode, once decoded from Base64)

- dg1: 0101
- dg2: 0102
- sod: 011D
- efcom: 011E

The following script gives an example of a client implementation of the solution, writing files to the disk (that can be read then by JMRTD):

```
\#!/usr/bin/perl


use utf8;
```

```
use JSON::XS;
use MIME::Base64;
use Data::Dumper;
use URI::Escape;


binmode STDOUT, ":utf8";
binmode STDIN, ":utf8";


sub bin2file {
my $filename = shift;
my $str = shift;
my $binmode=shift;
open OUT, ">$filename" or return undef;
binmode(OUT);
print OUT $str;
close OUT;
return 1;
}


my $cn=$ARGV[0];
my $serial=$ARGV[1];


$cn=uri_escape($cn);


print "https://localhost/devpki/?direct=1&action=mrtd&cn=$cn&serial=
$serial\n";
my $json=`wget --no-check-certificate --user=<admin> --
password=<passwd> -O -
"https://<pkihost>/pki/?direct=1&action=mrtd&cn=$cn&serial=
$serial" 2>/dev/null`;


my $jsonh=decode_json($json);


my $dg1=$jsonh->{response}->{dg1};
my $dg2=$jsonh->{response}->{dg2};
my $efc=$jsonh->{response}->{efcom};
my $sod=$jsonh->{response}->{sod};


bin2file("0101.bin",decode_base64($dg1));
bin2file("0102.bin",decode_base64($dg2));
bin2file("011D.bin",decode_base64($sod));
bin2file("011E.bin",decode_base64($efc));
```