

# Delphi-Windows Digital Signature Library Interface

Joseph Snyder  
SW Process Engineer  
snyderj@osehra.org

## Purpose

The purpose of these three files is to provide the interface between Delphi executables and the built-in Windows security functions. The push to supplement the traditional sign-on with the usage of a Personal Identity Verification (PIV) card is rapidly gaining momentum. One major focus of using these PIV cards is to further secure the prescribing of controlled substances.

There are existing versions of these interface files. One version was released in the FOIA copy of the Computerized Patient Record System (CPRS) for the Vista EHR. However, it was determined that the license for those files was incompatible with the license that the OSEHRA Vista uses for its repository. The files contained here are written by blanking the three necessary files and generating function calls based upon what was missing. These files are released with the Apache 2.0 license to match the license for the OSEHRA Vista repository

## Code Walkthrough

The interface consists of three separate files:

- XlfMime.pas
- Wcrypt2.pas
- WinSCard.pas

Each file contains different functions corresponding to a different focus around acquiring and verifying the security certificate of the card holder.

### *WinSCard.pas*

WinSCard.pas contains the constant values and function signatures necessary to connect to a Smart Card reader. Once a connection has been established to a local Smart Card reader, it is able to access the information about the certificate contained on a supplied card. These functions are all found in the Wincard.dll library.

### *Wcrypt2.pas*

Once the user's certificate has been accessed from the PIV or Smart Card, the next step is to verify that the certificate is valid and corresponds to a trusted user. The functions found in Wcrypt2.pas are to be used to open a certificate store and perform various checks to verify that the user has the correct identity and permissions. The functions in this file are called from one of two dlls: crypt32.dll or Advapi32.dll.

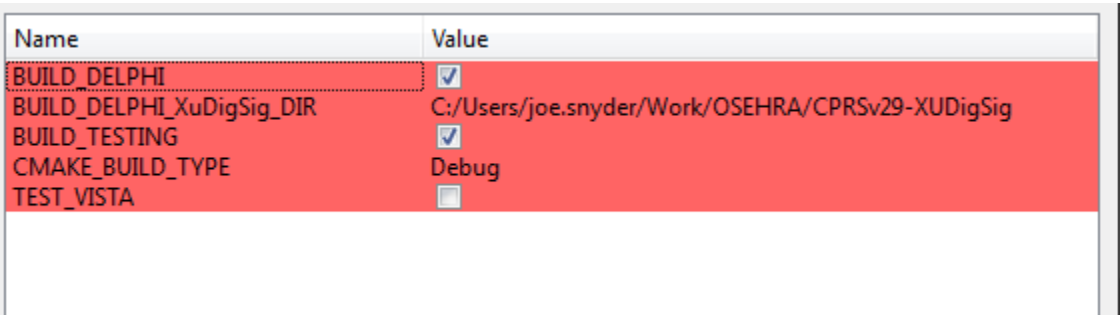
## *XlfMime.pas*

The functions found in the XlfMime.pas file are used to encode information for transmission between two sites. It uses the EncdDecd library with functions to accept both arbitrary string input and a stream of data from other sources. This library encodes and decodes the information using a Base64 encryption.

### Installation

These files were created to replace files from the FOIA release of the CPRS executable, so there is already a designed path to include these files. Please be aware that these variables and options will only be shown on a system with the proper Delphi environment.

To use these files with the OSEHRA Vista repository to build the CPRS executable, one would only need to set the "BUILD\_DELPHI\_XuDigSig\_DIR" variable in the CMake GUI to the path to a directory with these files. An example is found in the image below:



The image shows a screenshot of the CMake GUI's 'Cache' window. The window has two columns: 'Name' and 'Value'. The 'BUILD\_DELPHI\_XuDigSig\_DIR' variable is highlighted with a red dashed box. The 'Value' column for this variable shows the path 'C:/Users/joe.snyder/Work/OSEHRA/CPRsv29-XUDigSig'. Other variables visible include 'BUILD\_DELPHI' (checked), 'BUILD\_TESTING' (checked), 'CMAKE\_BUILD\_TYPE' (Debug), and 'TEST\_VISTA' (unchecked).

Name	Value
BUILD_DELPHI	<input checked="" type="checkbox"/>
BUILD_DELPHI_XuDigSig_DIR	C:/Users/joe.snyder/Work/OSEHRA/CPRsv29-XUDigSig
BUILD_TESTING	<input checked="" type="checkbox"/>
CMAKE_BUILD_TYPE	Debug
TEST_VISTA	<input type="checkbox"/>

This information would then be propagated to the Delphi project file upon running the CMake configure and generate steps.

These files are obviously not limited to just the CPRS application. To install the files into any other Delphi project, one would follow the standard method of including the files via the 'uses' directive. The files should be placed into a directory and added to a 'uses' subheading in the Delphi project file (\*.dpr). Again, a small code snippet example follows:

```
Uses
  ShareMem,
  Forms,
  winHelpViewer,
  ORSystem,
  wcrypt2 in 'C:/Users/joe.snyder/work/OSEHRA/CPRsv29-
XUDigSig\wcrypt2.pas',
  winSCard in 'C:/Users/joe.snyder/work/OSEHRA/CPRsv29-
XUDigSig\winSCard.pas',
  xlfMime in 'C:/Users/joe.snyder/work/OSEHRA/CPRsv29-
XUDigSig\xlfMime.pas',
<snip>
```

### How to Test

There are DUnit tests written for the three files which included in this download:

- UTXlfMime.pas
- UTWcrypt2.pas
- UTWinSCard.pas

### *Note about testing:*

One of the Wcrypt2 tests has multiple pass conditions based upon the return of the function being tested. The TestCertVerifyRevocation function has two passing conditions:

- the CertVerifyRevocation function returns **true** and has **no error** expressed
- The CertVerifyRevocation function returns **false** and the dwError value is set to '2148081683'.
  - This corresponds to a CRYPT\_E\_REVOCATION\_OFFLINE error as expected.

If the dwError value is any other value, the test will fail. Due to the absence of a revocation server in the testing environment, there is no dashboard result with the test returning true with no errors.

### *Integration of Testing*

#### OSEHRA Vista Repository

These files, much like the source files, have a place in the OSEHRA Vista repository. To test the files with the CPRS build process, place the files into the Packages\Order Entry Results Reporting\CPRS\Testing\Tests directory. Then, you would add an entry for each file under the 'uses' header to the CPRSTesting.dpr.in:

```
uses
  UTXlfMime in '@SOURCE@\Tests\UTXlfMime.pas' {UTXuDsigS},
  UTWcrypt2 in '@SOURCE@\Tests\UTWcrypt2.pas' {UTXuDsigS},
  UTWinSCard in '@SOURCE@\Tests\UTWinSCard.pas' {UTXuDsigS};
```

After running a Configure, Generate, and make, you can execute all of the tests at once by running the executable or by using the CTest program.

Running 'ctest -n' will show the available tests:

```
C:\Users\joe.snyder\work\OSEHRA\Vista-delphi-dev>ctest -N
Test project C:/Users/joe.snyder/work/OSEHRA/Vista-delphi-dev
Test #1: DUnitUTSignonCnf
Test #2: DUnitUTWcrypt2
Test #3: DUnitUTWinSCard
Test #4: DUnitUTXlfMime
```

Total Tests: 4

Then, the tests can be run together by executing 'ctest' or individually with a command like "ctest -R wcrypt2". The tests run will be determined by matching the test names against the string after the "-R".

An earlier attempt of the integration of the testing code can be found in the OSEHRA Gerrit instance:

<http://review.code.osehra.org/#/c/633/> .

## External project

For testing with DUnit without using the OSEHRA Vista repository, the files can be included into any existing DUnit testing setup. If one does not exist, the OSEHRA version is a good example to see how to generate a separate testing executable to run. The Delphi project directory which generates the OSEHRA testing can be found in the Packages/Order Entry Results Reporting/CPRS/Testing directory.

## Coverage Calculations

The testing of the code reports that 95% of the executable code is covered by the three test files. The coverage was calculated using the Delphi Code Coverage tool (<https://code.google.com/p/delphi-code-coverage/>) with the OSEHRA Vista testing setup.

The command used to generate the necessary HTML files for CTest to parse is:

```
$ ~/Downloads/CodeCoverage_win32_1.0_RC9/CodeCoverage.exe -e  
~/Work/OSEHRA/Vista-delphi-dev/CPRS/Testing/Bin/Debug/CPRSTesting.exe  
-html -u xlfMime winSCard wcrypt2 -sp ~/Work/OSEHRA/CPRsv29-XUDigSig/
```

## Future Work

The future of these three files does have a clear path for additional development. The files do not contain the signatures for every possible Windows function. As a new functionality is developed for these Delphi libraries, calls to other Windows functions will become necessary. These functions and any necessary data structures will be included as needed.