# PKCS11 Load Test Tool

## OVERVIEW

Many Government agencies and commercial organizations use PKI security tokens or smartcards that utilize various proprietary token or smartcard PKI implementations and/or openly accredited standards such as FIPS-201 (PIV).

For US Government agencies FIPS-201 is normally mandatory. Both FIPS-201 and virtually all proprietary and standards based PKI implementations implement PKCS#11 at the operating system or middleware API level. This is essential for interoperability. This API is more commonly called Cryptoki, pronounced crypto-key and short for cryptographic token interface

There has been little work done on tools to load test the entire PKI channel, which seems to be a missing test tool given the complexity of the channel, which typically looks like;

> Calling Application > Middleware or Windows Cryptoki > OS > PC/SC > USB > Reader > Token Interface > JavaCard VM > FIPS-201 JC Applet > Hardware Crypto Engine and memory > Applet > JCVM > Token Interface > Reader > USB > PC/SC > OS > Cryptoki > Calling Application

There are clearly many areas for issues in the channel above, and historically it has been difficult to find problems such as memory leaks at the various software and hardware levels. There is also the potential that some smartcards or tokens may have issues that may reduce the useful life of the tokens well below the stated MTBF. It is also possible that the security tokens may fail due to ‘false positives’ in their tamper prevention mechanisms. Finally, it is useful to be able to test PKCS#11 implementations under simulated load for performance purposes (i.e. against a HSM or Remote Desktop instance via a LAN/WAN network).

This simple test tool has therefore been developed to perform a common PKI cryptographic use-case using the Cryptoki PKCS#11 API many times in quick succession, providing a ‘load test’ environment that will validate or assist in identifying both integrity and performance issues by inducing failures quickly.

The test sits at the application level and as such exercises the entire channel other than the "Calling Application" in conjunction with PKCS#11 middleware. It exercises authentication, encryption and digital signature functionality. Other testing would be necessary to exercise the various possible applications (email, acrobat, word) that would call Cryptoki.

The test tool can support multiple USB connected smartcard readers and cards/tokens in a "round robin". With a round robin of two (2), 20,000 transactions can be reached with most cards in 3 days of continuous testing. The tool is tested with up to 8 cards/readers in the round robin.

## SCOPE

The scope of this test tool is as follows:

1) Performs load testing against a PKCS#11 token to provide support in validating reliability and integrity problems with existing implementations. The PKCS#11 commands are a simple subset of the total PKCS#11 functionality.

    a. Initialise a PKCS#11 library (using a configurable module path)
    b. Detect the presence of all PKCS#11 slots and list the available tokens.
    c. Xxxxx  kim – insert get card CPLC etc
    d. Perform a series of typical transactions against all available tokens, [n] number of times.  One transaction consists of the following PKCS#11 operations:

        i. Open Session (Read-only session)
        ii. Login (Normal User)
        iii. Query Objects (Find Private Key)
        iv. Query Objects (Find Public Key)
        v. Generate Random Data
        vi. Encrypt Block (RSA PKCS)
        vii. Digest (SHA-1)
        viii. Generate Signature (RSA PKCS)
        ix. Verify Signature (RSA PKCS)
        x. Decrypt (RSA PKCS)
        xi. Logout
        xii. Close Session

2) Generate a simple set of reporting outputs in CSV format that record the IC Serial Number, the output status and data of the above operations. Failures will be logged and the operator prompted to proceed.

## SCOPE EXCLUSIONS

- The initialisation and personalization of PKCS11/PIV tokens
- Broad testing of all PKCS11 functionality on a token
- Broad testing of support for compatibility with commercially available PKCS11 tokens or library modules (Currently, the Charismatics Middleware 1.1 and several FIPS-201 compliant cards have been provided).

## INSTALLATION

This test tool is a self-contained Windows executable, and as such does not need installation as such. However, it requires the following dependent components to be installed and working:

1) Microsoft Windows 7 or above (64-bit)
http://www.microsoft.com/windows

2) Microsoft Visual C++ 2010 SP1 Redistributable Package (X64)
http://www.microsoft.com/en-au/download/details.aspx?id=13523

3) A PC-SC compliant smart-card reader

4) A suitable PKCS#11 Library

## OPERATION

The test tool operates as a command-line executable, so familiarity with the DOS-style command-prompt is recommended. By default, the test tool is named 'PKCS11LoadTest.exe'.

The application has several command-line arguments that configure the behaviour of its operation. To see a listing of these command-line parameters from within the application, including a description of each, run the executable with the '-H' parameter (for help).

The command-line parameters are as follow:
PKCS11LoadTest [-D] -L <Library> -P <Pin> [-C Count] [-I Interval] [-H]

| PARAMETER | DESCRIPTION |
|---|---|
| -L | REQUIRED - The full or relative path to the PKCS#11 Library Module (See the PKCS#11 standard for more information).<br><br>Example: '-L C:\Windows\System32\cmp11.dll' |
| -P | REQUIRED - The ASCII/Numeric PIN number for the active tokens.<br><br>Example: '-P 11111111' |
| -C | The number of simulated transactions to perform in this session.<br><br>Example: '-C 10'<br>Default: 10 |
| -I | The amount of time to wait between transactions in milliseconds'<br><br>Example: '-I 1000'.<br>Default: 1000 (1 second) |
| -D | If specified, the application will produce verbose debug information to assist in diagnosing issues.<br><br>*NOTE: If you want to use this flag, it is recommended that you pass it first so that it takes effect immediately, even before further options processing.* |
| -H | Displays the help message and exits. |

At a minimum, the PKCS11 module path, User PIN and Key identifier must be supplied, for example:

*X:\PKCS11LoadTest.EXE –L cmp11.dll –P 11111111 –K 9C07*

NOTES:

- When running multiple cards in a "round robin" the PIN for all cards must be the same.
- A separate '.log' file will be created for each card, using the CPLC IC Serial Number as the name. These will always be appended to so you will need to delete previous files manually if you wish to start from a clean file.
- The log file format is an ASCII Comma-Separated Value (CSV) file and has the following format:
  *TIMESTAMP,SERIAL,ITERATION,OPERATION,OUTCOME[,DATA]<CRLF>*

WARNING – Because this test tool cycles transactions very fast, if an incorrect PIN is used it is likely that the token will be locked before the operator has timed to respond. In some scenarios this may make the token un-recoverable due to excess failed PIN attempts or if the System Operator (SO) PIN cannot be obtained.

## DEVELOPMENT

The test tool is written in C++, using Microsoft Visual Studio 2010 Professional Edition - SP1 as the development environment. Aside from the runtime requirements above, there are no special installation steps necessary.

To open the project, simply click on the PKCS11LoadTest.vcxproj file or PKCS11LoadTest solution.

This project is hosted using Google Code at the following repository location:
https://code.google.com/p/pkcs11-load-test/

This project can be checked out anonymously, using the following subversion command:

*svn checkout **http**://pkcs11-load-test.googlecode.com/svn/ pkcs11-load-test-read-only*

## LICENSE

This application is licensed under the MIT License, a copy of which is provided below:

*The MIT License (MIT)*

*Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:*

*The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.*

*THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.*

## REFERENCES / STANDARDS

| | |
|---|---|
| PKCS#11 | http://www.emc.com/emc-plus/rsa-labs/standards-initiatives/pkcs-11-cryptographic-token-interface-standard.htm<br><br>*NOTE: This has been tested against library modules implementing PKCS#11 v2.11 and v2.20 only.* |
| PCSC | http://pcscworkgroup.com/ |