FLASH, a fast asymmetric signature scheme for low-cost smartcards

Implementation

Jacques Patarin, Nicolas Courtois, Louis Goubin

Bull CP8
68 route de Versailles - BP 45
78431 Louveciennes Cedex
France
J.Patarin@frlv.bull.fr, courtois@minrank.org, Louis.Goubin@bull.net

1 Introduction

Implementation is supllied in the form of C++ program that uses Victor Shoup's NTL library.
NTL provides state of the art algorithms for factoring polynomials over finte fields.
The implementation is (apparently) written in ANSI C++.
The source is now tested to compile and work correctly with respect to all tests on the following platforms:
1. Under Windows 9x with Microsoft Visual Studio (we provide the project file NessHfe.dsp).
2. Under Linux/g++ (with Makefile).
Both Linux/Windows cases work OK on a Pentium processor.
3. The source is known not to be portable on a big-endian machine (e.g. sparc) This should be addressed in ulterior versions
Authors wish to thank Louis Granboulan for extensive help.

2 Principle of the implementation

The general philosophy of the program is the following:
The main file is NessHfe.cpp.
There is a single executable that will be called NessHfe.exe or so (system and compiler-dependent).
This file is used in a command-line way.
The output directory in windows should be "C:\Program Files\Multivariate Signature".

Important: The executable NessHfe.exe must be lauched in the directory containing the public or secret key described later.

The program implements potentially many multivariate schemes in such a way that the main program remains the same, and all the algorithm-dependent information for signature generation and verification is stored in a file written in a standard way and that contains not only the public or secret key but also a description of the algorithm.

The public key should be a file with extension .PKey. It is done according to a standard described in the document PKey.ps.
The secret key should be a file with extension .SKey. It is done in a less standart way that is not published, and is always encrypted with RC6.
This philosophy of design allows pro-active approach to the signature: we may add a new algorithm without changing the implementation.
Let || be the string concatenation operation.

2.1 Generation of a pair of public/secret key

To generate a pair of public/secret key we first chose an Id string S with at most 8 charactes or numbers, for example S="Nessie". We write a following command:
NessHfe.exe setup Flash S
The program will ask for a long string that is hashed with SHA-1 and supplied for NTL's random number genrator (using MD5).
This part takes up to few minutes on a PC and since it is done only once in the lifetime of the signature, it has not been fully optimized yet.
Two files denoted S||".Pkey" and S||".Skey" will be written in the working directory of the program (for example Nessie.PKey and Nessie.SKey). The last is encrypted.

2.2 Generation of a signature

To generate a signature we write a command:
NessHfe.exe S sign FileNameWithOptionalFullPath
This requires the presence of S||".Skey" in the current directory, and the knowledge of the password.
Every file is treated as a binary file and one must be careful about comparing results of signatures of a file.
The signature is displayed and also written to certif.txt. It is deterministic.

2.3 Verification of a signature

To verify a signature we write a command:
NessHfe.exe S check FileNameWithOptionalFullPath
This requires the presence of S||".Pkey" in the current directory.
The program displays if the signature is valid or not, and it returns 1 if invalid and 0 if valid. If something wrong happens, a different return code is returned, -1 is when command line parameters are not correct.

3 Test vectors

Made with our original windows executable included in .\windows\

3.1 Tests on key generation

Parameters used in our test value:

- The password for all keys (it is used later for other tests) must be 0123456789
- The e-mail adress must be entered as default "submissions@cryptonessie.org" (case sensitive)

- The random string must be the default "40db189e97485c3d9a5d5ca11246e49b1c3ad065"
- The key number must be 0 (the default value)

The resulting files *.SKey and *.PKey must be identical to ours.
The test is done on 3 following examples (files generated in those 3 are used for the following verifications).

```
-----------------------------------------------------------------
Command line:
NessHfe.exe setup Flash Flash
The resulting files are Flash.SKey (Md5=5b8f7f9ad4bdc536fedd1f06bacdf4b0)
and Flash.PKey (Md5 will be different at each time because
the public key contains the time at which it was generated).
```

3.2 Tests on signatures

They contain 3 pairs of public/secret keys, 3 files to test signatures on. The password for all keys is 0123456789

```
-----------------------------------------------------------------
Comand line:
NessHfe.exe Flash sign check.txt

Resulting signature:
27ac14dc5d41f8b839b76e1167daf5c0b018254838d5c77a5a084cd2bfdae1f1b451bad7aa

Check with command line:
NessHfe.exe Flash check check.txt 27ac14dc5d41f8b839b76e1167daf
5c0b018254838d5c77a5a084cd2bfdae1f1b451bad7aa

-----------------------------------------------------------------
Comand line:
NessHfe.exe Flash sign test.txt

Resulting signature:
b3cb3e5d2862d0efa2d512529655e5db7d0621f1b626ad5048bdbc4e8ae7de5e406b995c1f

Check with command line:
NessHfe.exe Flash check test.txt b3cb3e5d2862d0efa2d512529655e5
db7d0621f1b626ad5048bdbc4e8ae7de5e406b995c1f

-----------------------------------------------------------------
Comand line:
NessHfe.exe Flash sign  verify.txt

Resulting signature:
35f4a5d8424b452b67d77e7edfad729b4980851e14489b5285dae573
aaabc12c179615946b
```

```
Check with command line:
NessHfe.exe Flash check verify.txt 35f4a5d8424b452b67d77e7edfad729b4980851e14489b5285dae573aaab
```

4 Practical working with Windows

The working directory in windows should be
"C:\Program Files\Multivariate Signature"
The executable filename must be "C:\Program Files\Multivariate Signature\NessHfe.exe"
Double-click Install.reg to register the program !
The three .bat files supplied should also be in "C:\Program Files\Multivariate Signature". Use them to generate keys.
When we right-click on a file, we have a possibility to sign/check signatures for the FLASH scheme.