



Open NFC - Security Stack - AC File Generator Tool

Document Type:	Functional Specification
Reference:	FRS_NFC_1104-244 Version 0.2 (10545)
Release Date:	April 27, 2011
File Name:	FRS_NFC_1104-244 Open NFC - Security Stack - AC File Generator Tool v0.2.pdf
Security Level:	General Business Use

Disclaimer

This document is licensed under the Creative Commons Attribution 3.0 license (<http://creativecommons.org/licenses/by/3.0/>). (You may use the content of this document in any way that is consistent with this license and if you give proper attribution (<http://www.open-nfc.org/license.html#attribution>)).

Copyright © 2011 Inside Secure

Open NFC and the Open NFC logo are trademarks or registered trademarks of Inside Secure.

Other brand, product and company names mentioned herein may be trademarks, registered trademarks or trade names of their respective owners.

History

Version	Date	Comments
0.1	April 20, 2011	Draft
0.2	April 27, 2011	First Release

References

[FRS-SEC] FRS_NFC_1104-241 Open NFC - Security Stack v0.4.pdf (12699)

[SE-
PKCS15] STS_NFC_1104-240 Open NFC - PKCS15 Applet 1.0 Specification v0.4.pdf
(10545)

Summary of Contents

1	Introduction.....	6
1.1	Acronyms.....	6
1.2	Notations.....	6
2	Command Line.....	7
2.1	Invoking the tool.....	7
2.2	Command Line Options.....	7
3	Source File Format.....	9
3.1	XML Syntax.....	9
3.2	XML Field Description.....	10
3.2.1	Path.....	10
3.2.2	Access Control Index Element.....	10
3.2.3	Access Control Element.....	10

1 Introduction

This document describes the "ACL Generator Tool", a PC tool designed to generate the ACL binary stream to be written into the EF(SE-ACF) file of "PKCS#15 Application" of the SE.

1.1 Acronyms

Acronym	Meaning
ACE	Access Control Entry
ACIE	Access Control Index Entry
ACF	Access Control File
ACL	Access Control List
AID	Application identifier (ISO/IEC 7816)
APDU	Application Protocol Data Unit
ASN.1	Abstract Syntax Notation 1
DER	Distinguished Encoding Rules
EF	Elementary File (ISO/IEC 7816-4)
FID	File identifier (ISO/IEC 7816-4)
JRE	Java Runtime Environment
PKCS	Public-Key Cryptography Standards
SE	Secure Element
SHA-1	Secure Hash Algorithm 1
XML	Extensible Markup Language

1.2 Notations

All hexadecimal values shall be written between quotes. For instance, the value '64' denotes the decimal value 100.

2 Command Line

The ACL Generator Tool is a command line Java™ application.

A JRE version 6 or higher is required.

2.1 Invoking the tool

First ensure that `java.exe` is in the path, or specify the full path to `java.exe`.

```
C:\>java -jar aclgen.jar <options>
```

where `<options>` are described in section 2.2.

2.2 Command Line Options

```
C:\>java -jar aclgen.jar -?
```

```
Usage: java[.exe] -jar aclgen.jar [<options>] <command> [<flags>]
```

where possible options include:

```
-h -?      Print a synopsis of possible options and exit  
-version   Print version information and exit  
-q         Do not print messages
```

where possible commands include:

```
-gen <file.xml> Generate an ACF image from the specified XML file  
this command has the following flags:  
-o <file> Specify the name of the output binary file  
-s <size> Specify the output file size  
-f:1       Specify output format (ACIE before ACE) - default  
-f:2       Specify output format (ACE before ACIE)  
-linux     Generate a principal value corresponding to a UID/GID  
this command has the following flags:  
-uid <uid> Specify the UID value  
-gid <gid> Specify the GID value  
-hash <file> Generate a hash value corresponding to <file>
```

Generation of an ACF File

The `-gen` command generate an ACF image from the specified XML file.

If the `-o` option is not specified, the default output binary file is the name of the source file where the extension (if any) is replaced with the `.bin` extension.

If the `-s` option is specified, the output binary file is padded with an Empty element to generate a file of the requested size.

The `-f:1` (default) and `-f:2` options can be used to generate variants of the output binary file. The `-f:2` option should generate a smaller file size.

Generation of a Principal Value for Linux

The `-linux` command generates a principal value corresponding to a UID/GID. The flags `-uid` and `-gid` specify the value of the UID and GID. The principal value is generated on

stdout as list of hexadecimal values. This function may be used to generate the default principal value(s) for the Security Stack configuration.

Generation of a Principal Value from a Hash value

The `-hash` function generate a hash value from the specified file (a X509 certificate or an Android package APK). The principal value is generated on stdout as list of hexadecimal values. This function may be used to generate the default principal value(s) for the Security Stack configuration.

3 Source File Format

The ACL Generator Tool parses source files containing the following XML description of the EF(SE-ACF) file.

3.1 XML Syntax

The example just below is intended to be self-explanatory (texts in bold are placeholders for actual values).

```
<?xml version="1.0"?>
<acl format="1.0">
  <!-- Optional: Path of the EF(SE-ACF) file -->
  <path>0X50 0X36</path>

  <!-- ACIE with no AID, ie, applying to all SE applications -->
  <acie ace="idAnyAID" />

  <!-- ACIE with a first AID -->
  <acie ace="idMyAppAID">
    <aid>0X?? 0X?? 0X?? 0X?? 0X?? 0X?? ... 0X??</aid>
  </acie>

  <!-- ACIE with another AID, sharing the ACE of the first AID -->
  <acie ace="idMyAppAID">
    <aid>0X?? 0X?? 0X?? 0X?? 0X?? 0X?? ... 0X??</aid>
  </acie>

  <!-- ACE granting all permissions to developers -->
  <ace id="idAnyAID">
    <principals>
      <principal type="hash">
        0X?? 20-byte hash of the developer certificate 0X??
      </principal>
    </principals>
    <!-- tag permissions is not present: all permissions are granted -->
  </ace>

  <ace id="idMyAppAID">
    <principals>
      <principal type="hash">
        0X?? 20-byte hash of application#1 certificate 0X??
      </principal>
      <principal type="x509">
        //home/test/my_certificate.der
      </principal>
      <principal type="apk">
        //home/test/my_application.apk
      </principal>
      <principal type="linux" uid="123" gid="456"/>
    </principals>
    <permissions>
      <permission type="APDU">
        <header>0X?? 0X?? 0X?? 0X??</header>
        <mask>0X?? 0X?? 0X?? 0X??</mask>
      </permission>
    </permissions>
  </ace>
```

</acl>

3.2 XML Field Description

The `acl` tag contains an unordered sequence of the following tags:

- An optional `path` tag
- Several (possibly none) `ace` tags
- Several (possibly none) `acie` tags

3.2.1 Path

The `path` tag contains the FID (most significant byte first) of the EF(SE-ACF) file, encoded as an hexadecimal string. It is optional. Its default value is '5036'.

3.2.2 Access Control Index Element

The `acie` tag associates an AID with an access control policy (ACE).

- A missing AID indicates that the ACE applies to all applications of the SE.

The `ace` attribute of the `acie` tag is mandatory and must take the value of an `id` attribute of an existing `ace` tag.

The `aid` tag is optional. If specified, its value must be an hexadecimal string with a length in the range 5 to 16.

3.2.3 Access Control Element

The `ace` tag describes the permissions assigned to a list of principals.

- A missing or empty `principals` tag indicates that permissions apply to all terminal applications.
- A missing or empty `permissions` tag indicates that all permissions are granted.

The `id` attribute of the `ace` tag is mandatory. Two `ace` tags cannot have the same `id`. An `ace` tag must be referred to from at least one `acie` tag.

The `principals` tag of the `ace` tag is optional. If specified, it contains a possibly empty list of `principal` tags that must have a `type` attribute with the value "hash", "x509", "linux" or "apk".

If the `type` is "hash", the value of the `principal` tag must then be the 20-byte hexadecimal string of the hash of the certificate that was used to sign the Android application.

If the `type` is "x509", the value of the `principal` tag must then be the path of an X509 certificate encoded with DER. The tool will compute the SHA-1 hash value of the certificate.

If the `type` is "linux", the `principal` tag shall contain an attribute "uid" with the value of the UID and/or an attribute "gid" with the value of the GID. The tool will compute the principal value corresponding to the UID and/or GID values.

If the type is "apk", the value of the `principal` tag must then be the path of an application package or Android (APK) certificate encoded with DER. The tool will extract the X509 certificate from the APK file and compute the SHA-1 hash value of the certificate.

The `permissions` tag of the `ace` tag is optional. If specified, it contains a possibly empty list of `permission` tags that must have a `type` attribute with the "APDU" value. The value of the `permission` tag must then contains two mandatory tags, `header` and `mask`, that must both contain a 4-byte hexadecimal string.