Future Technology Devices International Ltd

SafeGuard-IT Programmer's Guide

Document Reference No. Doc No FT__0001 Version 1.0

Issue Date: 2007-07-20

Future Technology Devices International Ltd (FTDI)

373 Scotland Street, Glasgow G5 8QB United Kingdom

Tel.: +44 (0) 141 429 2777 Fax: + 44 (0) 141 429 2758

E-Mail (Support): support1@ftdichip.com Web: http://www.ftdichip.com

Neither the whole nor any part of the information contained in, or the product described in this manual, may be adapted or reproduced in any material or electronic form without the prior written consent of the copyright holder. This product and its documentation are supplied on an as-is basis and no warranty as to their suitability for any particular purpose is either made or implied. Future Technology Devices International Ltd will not accept any claim for damages howsoever arising as a result of use or failure of this product. Your statutory rights are not affected. This product or any variant of it is not intended for use in any medical appliance, device or system in which the failure of the product might reasonably be expected to result in personal injury. This document provides preliminary information that may be subject to change without notice. No freedom to use patents or other intellectual property rights is implied by the publication of this document. Future Technology Devices International Ltd, 373 Scotland Street, Glasgow G5 8QB United Kingdom. Scotland Registered Number: SC136640

Table of Contents

1 Introduction

The FT232R USB UART and FT245R USB FIFO IC devices from Future Technology Devices International Ltd. incorporate the FTDIChip-ID™ security dongle feature. The FTDIChip-ID is a unique, 32-bit number that is burned into the device during manufacture and is readable over USB, thus forming the basis of a security dongle that can be used to protect customer application software from being pirated. Neither product manufacturers nor end users can change the number by any means.

The simplest implementation of software security using the FTDIChip-ID feature would be one in which the software being protected reads the FTDIChip-ID from a security dongle at startup and only continues to operate if the correct value is read. The obvious problem with this approach is that every copy of the application software sold must be individually compiled with the FTDIChip-ID value that is programmed into the security dongle sold with the software.

Using the SafeGuard-IT security ActiveX control, a software package need only be compiled once and only two additional files (the public key and DLL) need be distributed with the application software.

2 Method

The protection scheme utilised by SafeGuard-IT is known as asymmetric cryptography since it uses both public and private keys. This is a form of encryption that has been in use since the mid-1970s and is becoming an increasingly popular method of securely transmitting data via the Internet.

The first step in utilizing this system involves generating the public and private keys. A call to the GenerateKey() function creates the key pair. These keys are then saved to the hard drive as binary files. While the public key is distributed with the software package, the private key needs to be kept by the developer in a secure location.

Next, a small data packet is generated that is comprised of information from the private key and the FTDIChip-ID. This data packet is stored in the User Area memory of the security dongle by calling the SignDongle() function. The process of writing/verifying this data packet to a security dongle (known as "signing") is the final step in preparing a dongle for deployment.

Security dongles can be signed both with and without an additional password. Dongles can be signed with a password using the SignDonglePassword() function.

Two additional functions are provided for verifying the presence of a correctly signed security dongle: VerifyDongle() and VerifyDonglePassword(). These functions are compiled into the user application software and can either be called at program startup to verify the presence of the security dongle or multiple times during operation of the software to verify that the dongle has not been removed. Note that the public key corresponding to the private key used to sign the dongle is required to use these functions successfully. If a password was used to initially sign a security dongle, then it must be used every time the VerifyDonglePassword() function is called. It is up to the application developer to decide whether the operator should be prompted for the password every time the function is called.

The methods GenerateKey(), SignDongle() and SignDonglePassword() and the PrivateKey property are all accessible via the ISetupDongle interface. The methods VerifyDongle() and VerifyDonglePassword() are accessible via the IUseDongle interface.

The public key and SafeGuard-IT DLL file must be installed along with the application software. In addition, the SafeGuard-IT DLL must be registered using the regsvr32 utility. Most modern software installation utilities can automate this process.

NOTE: The use of the SafeGuard-IT ActiveX control requires FTDI's CDM drivers to be installed for the device.

3 SafeGuard-IT ActiveX Control

3.1 SetupDongle Interface

3.1.1 SetupDongle Methods

The methods accessible via the SetupDongle interface are used to prepare a dongle for use. Once the dongle has been programmed using these methods, the dongle can be used with the UseDongle interface methods to ensure security. The PrivateKey property is needed for preparing the dongle for use. The corresponding *PublicKey* value is required for using the dongle.

3.1.1.1 GenerateKey

Description
Generates a new key pair to use for signing and validating dongles. The public part of the key is returned and the private part of the key is stored in the PrivateKey property.

HRESULT GenerateKey ([out, retval] SAFEARRAY(byte) * *PublicKey*)

Parameters

PublicKey [out] A pointer to the PublicKey value is returned.
The PrivateKey value is accessible via the PrivateKey property after this method has been executed.

Return Value
HRESULT value, as defined in winerror.h.

Remarks
Both the PublicKey and PrivateKey values should be saved to binary files for use with the SignDongle, SignDonglePassword, VerifyDongle and VerifyDonglePassword methods. The *PublicKey* value will be required to decrypt dongles signed with the *PrivateKey* value.

3.1.1.2 SignDongle

Description
Signs all attached dongles using the private key in the PrivateKey property.

HRESULT SignDongle ()

Parameters
None.

Return Value
HRESULT value, as defined in winerror.h.

Remarks
The PrivateKey property must be assigned before this function can be called. The actual signature placed on the dongle is based on the dongle's FTDIChip-ID, encrypted with the private part of the key. The public part of the same key must be used to verify the dongle. All dongles connected to the PC will be signed when this method is executed.

3.1.1.3 SignDonglePassword

Description
Signs all attached dongles with *Password* using the private key in the PrivateKey property.

HRESULT SignDonglePassword ([in] BSTR *Password*)

Parameters
Password A password string used for encryption of the dongle.

Return Value
HRESULT value, as defined in winerror.h.

Remarks
The PrivateKey property must be assigned before this function can be called. The actual signature placed on the dongle is based on the dongle's FTDIChip-ID, encrypted with the private part of the key. The public part of the same key must be used along with the same password to verify the dongle. All dongles connected to the PC will be signed when this method is executed.

3.1.2 SetupDongle Properties

3.1.2.1 PrivateKey

Description
Gets or sets the *PrivateKey* property that is used for signing dongles.

Get PrivateKey Property
[propget] HRESULT PrivateKey ([out, retval] SAFEARRAY(byte) * *pVal*)

Parameters
pVal A pointer to the *PrivateKey* value currently in use for signing dongles.

Return Value
HRESULT value, as defined in winerror.h.

Set PrivateKey Property
[propput] HRESULT PrivateKey ([in] SAFEARRAY(byte) *newVal*)

Parameters
newVal The *PrivateKey* value to be used for signing dongles.

Return Value
HRESULT value, as defined in winerror.h.

Remarks
The *PrivateKey* property should be used after executing the GenerateKey method to get the *PrivateKey* value which should then be saved to a binary file.
The *PrivateKey* property should also be used to set the *PrivateKey* value, which can be read from a binary file, before executing the SignDongle or SignDonglePassword methods.

3.2 UseDongle Interface

3.2.1 UseDongle Methods

The UseDongle methods allow a programmed dongle to be verified. The *PublicKey* value is used to decrypt the dongle and determine if the dongle is valid. Note that the *PublicKey* value must match the *PrivateKey* value returned from GenerateKey() which was used to sign the dongle.

3.2.1.1 VerifyDongle

Description
Verifies a signed dongle with the passed *PublicKey* value.

HRESULT VerifyDongle ([in] SAFEARRAY(byte) *PublicKey*, [out,retval] VARIANT_BOOL* *Valid*)

Parameters
PublicKey The *PublicKey* required to decrypt the dongle.
Valid Pointer to the *Valid* value. True if the validation was successful or false if the validation was unsuccessful.

Return Value
HRESULT value, as defined in winerror.h.

Remarks
If the *PublicKey* used with this method does not correspond to the *PrivateKey* that was used to encrypt the dongle, *Valid* will be returned as FALSE. If **any** dongle connected to the PC is determined to be valid, *Valid* will be returned as TRUE.

3.2.1.2 VerifyDonglePassword

Description
Verifies a signed dongle with the passed *PublicKey* and *Password* values.

HRESULT VerifyDonglePassword ([in] SAFEARRAY(byte) *PublicKey*, [in] BSTR *Password*,
[out,retval] VARIANT_BOOL* *Valid*)

Parameters
PublicKey The *PublicKey* required to decrypt the dongle.
Password A password string used for decryption of the dongle.
Valid Pointer to the *Valid* value. True if the validation was successful or false if the validation was unsuccessful.

Return Value
HRESULT value, as defined in winerror.h.

Remarks
If the *PublicKey* used with this method does not correspond to the *PrivateKey* that was used to encrypt the dongle, *Valid* will be returned as FALSE. If the *Password* used with this method does not match the *Password* used with the SignDonglePassword method, *Valid* will be returned as FALSE. If **any** dongle connected to the PC is determined to be valid, *Valid* will be returned as TRUE.

4 Contact Information

Head Office - Glasgow, UK

Future Technology Devices International Limited
373 Scotland Street
Glasgow G5 8QB
United Kingdom

Tel: +44 (0) 141 429 2777
Fax: +44 (0) 141 429 2758
E-Mail (Sales): sales1@ftdichip.com
E-Mail (Support): support1@ftdichip.com
E-Mail (General Enquiries): admin1@ftdichip.com
Web Site URL: http://www.ftdichip.com
Web Shop URL: http://apple.clickandbuild.com/cnb/shop/ftdichip

Branch Office - Taiwan

Future Technology Devices International Limited (Taiwan)
4F, No 16-1, Sec. 6 Mincyuan East Road
Neihu District
Taipei 114
Taiwan, R.O.C.

Tel: +886 2 8791 3570
Fax: +886 2 8791 3576
E-Mail (Sales): tw.sales1@ftdichip.com
E-Mail (Support): tw.support1@ftdichip.com
E-Mail (General Enquiries): tw.admin1@ftdichip.com
Web Site URL: http://www.ftdichip.com

Branch Office - Hillsboro, Oregon, USA

Future Technology Devices International Limited (USA)
7235 NW Evergreen Parkway, Suite 600
Hillsboro, OR 97124-5803
USA

Tel: +1 (503) 547-0988
Fax: +1 (503) 547-0987
E-Mail (Sales): us.sales@ftdichip.com
E-Mail (Support): us.support@ftdichip.com
E-Mail (General Enquiries): us.admin@ftdichip.com
Web Site URL: http://www.ftdichip.com

Distributors and Sales Representatives
Please visit the Sales Network page of the FTDI Web site for the contact details of our distributor(s) in your country.