|  |  |
| --- | --- |
| doc-header-print-portrait |  |
|  | SecurityFramework Integration  Requirements |
| Version 6.0 |  |
| SecurityFramework Integration Requirements | |

Table of Contents

[Change History 1](#_Toc283991279)

[1 Introduction 2](#_Toc283991280)

[2 Crypto Integration 2](#_Toc283991281)

[2.1 General Requirements 2](#_Toc283991282)

[2.2 Password Management 2](#_Toc283991283)

[2.3 Crypto 3](#_Toc283991284)

[2.4 Data Protection API (DPAPI) 3](#_Toc283991285)

[2.5 CryptoSetup 3](#_Toc283991286)

[2.6 Unit Test 3](#_Toc283991287)

# Change History

This section contains the history of changes to this document.

|  |  |  |
| --- | --- | --- |
| Date | Author | Description of Change |
| 1/28/11 | Sudip Chakraborty | Initial Revision |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# 

# Introduction

This document specifies the requirements for integrating the SecurityFramework code base with the MetraNet implementation. At this point, the SecurityFramework code has been moved into the ~Source\MetraTech\SecurityFramework folder on the 6.4.0-MetraNetSF-Dev branch and the team is able to build the code.

# Crypto Integration

The current security implementation is in the ~Source\MetraTech\Security folder. The goal is to have one security module going forward. This will be the SecurityFramework module. To facilitate this, the code in the current security modules will be moved to the SecurityFramework codebase. Since there is a lot of existing MetraNet code that makes use of some of the classes in the current security code (eg. CryptoManager, PasswordManager, etc) and we don’t want to update the MetraNet code because of this migration – we will leave those classes where they are and their API’s will forward calls to the implementation in the SecurityFramework codebase.

## General Requirements

1. Existing MetraNet code that reference the current security classes under S:\MetraTech\Security must continue to work without any changes.
2. All new MetraNet code that need to use security functionality will call the SecurityFramework code directly.
3. The SecurityFramework code cannot have any dependencies on current security code. Hence, it cannot reference any of the following:
   * MetraTech.Security.dll
   * MetraTech.Security.Crypto.dll
   * CryptoSetup.exe
4. The assembly produced by the SecurityFramework code will be called MetraTech.SecurityFramework.dll

## Password Management

The following requirements must be satisfied for the password management functionality.

1. Existing clients of PasswordManager (S:\MetraTech\Security\PasswordManager.cs) and Auth (S:\MetraTech\Security\Auth.cs) must continue to work without any changes.
2. The API’s in PasswordManager and Auth must forward calls to the implementation in SecurityFramework.
3. New clients of the password management and auth functionality must be able to use the SecurityFramework directly. Provide sample code demonstrating this use.
4. The configuration file for password management (R:\config\security\mtpassword.xml) must stay the same.

## Crypto

The following requirements must be satisfied for the crypto functionality.

1. Existing clients of CryptoManager (S:\MetraTech\Security\Crypto\CryptoManager.cs) and CryptoInstall (S:\MetraTech\Security\Crypto\CryptoInstall.cs)must continue to work without any changes.
2. The API’s in CryptoManager and CryptoInstall must forward calls to the implementation in SecurityFramework.
3. New clients of the crypto functionality must be able to use the SecurityFramework directly. Provide sample code demonstrating this use.
4. The configuration file for crypto management (R:\config\security\mtsecurity.xml) must stay the same.

## Data Protection API (DPAPI)

The following requirements must be satisfied for the DPAPI functionality.

1. The DPAPI functionality will be migrated to the SecurityFramework code base.

## CryptoSetup

The CryptoSetup.exe tool must continue to work as it does now. The code for this tool will be migrated to the SecurityFramework code base. Since the new CryptoSetup code cannot reference MetraTech.Security.Crypto.dll, the code will have to call the corresponding crypto implementation in the SecurityFramework code.

## Unit Test

The unit test code for SecurityFramework will be placed under ~6.4.0-Development\UnitTests\MetraTech\SecurityFramework. It is assumed that the 6.4.0-Development directory is the parent of the Source directory (i.e. ~6.4.0-Development\Source\MetraTech).