

Lost Password Scheme V1.1

Overview

Version 1.1

12/29/2014

Prepared by  


[www.inforeliance.com](http://www.inforeliance.com)

Approval Page

Background

This document has been prepared for T2 – National Center for Telehealth and Technology

Point of Contact

The Point of Contact for questions about this document is:

Scott Coleman

InfoReliance Corporation

[Street address]

[City, State, zip]

Phone: ([xxx]) [xxx-xxxx]

Email: scott.coleman@tee2.org

Submitted By:

|  |  |
| --- | --- |
| This document is prepared by: | This document is endorsed by: |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Signature Date | Signature Date |
|  | [name], [title] |

Approved By:

|  |
| --- |
| This document was approved by: |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Signature Date |
| [customer\_name], [customer\_title] |

Document History

Revisions

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Action |
| 1.0 | 12/3/2014 | Scott Coleman | Created Document |
| 1.1 | 12/30/2014 | Scott Coleman | Added cross platform functionality |

Table of Contents

[1.0 Introduction 1](#_Toc407699755)

[1.1 Purpose 1](#_Toc407699756)

[2.0 Components 1](#_Toc407699757)

[3.0 Key Handling Overview 2](#_Toc407699758)

[3.1 Initialization 2](#_Toc407699759)

[3.2 Login/Data Access With Pin 2](#_Toc407699760)

[3.3 Login/Data Access with Security Answers 3](#_Toc407699761)

[4.0 Native Client Interaction with T2Crypto 3](#_Toc407699762)

[5.0 Cordova T2Crypto Module 1](#_Toc407699763)

[5.1 Introduction 1](#_Toc407699764)

[5.2 Globals 1](#_Toc407699765)

[5.2.1 T2Crypto.prototype.changePinUsingAnswers 2](#_Toc407699766)

[5.2.2 T2Crypto.prototype.changePinUsingPin 2](#_Toc407699767)

[5.2.3 T2Crypto.prototype.checkAnswers 3](#_Toc407699768)

[5.2.4 T2Crypto.prototype.checkPin 3](#_Toc407699769)

[5.2.5 T2Crypto.prototype.deinitializeLogin 4](#_Toc407699770)

[5.2.6 T2Crypto.prototype.getDatabaseKeyUsingPin 4](#_Toc407699771)

[5.2.7 T2Crypto.prototype.initializeLogin 5](#_Toc407699772)

[5.2.8 T2Crypto.prototype.initT2Crypto 5](#_Toc407699773)

[5.2.9 T2Crypto.prototype.isInitialized 6](#_Toc407699774)

[6.0 Cross Platform App Interaction With T2Crypto Cordova Module 6](#_Toc407699775)

Table of Figures

[Figure 1 Lost Password Scheme Components 1](#_Toc407699214)

[Figure 2 Key Handling Overview 2](#_Toc407699215)

[Figure 3 Demo App Class Diagram 4](#_Toc407699216)

[Figure 4 Initialization and Login with Pin 1](#_Toc407699217)

[Figure 5 Login with Answers and Change Pin 2](#_Toc407699218)

# Introduction

## Purpose

The purpose of this document is to outline a potential solution for providing lost password assistance for T2’s mobile apps. This solution is comprised of providing security questions as backup login. This paper outlines the technical requirements and design for such a system

# Components

This solution builds upon T2’s FIPS 140-2 solution as outlined in T2’s document FIPSDataStorage.docx. This document will refer to this solution simply as T2’s FIPS library (as implemented in the code library libcrypto).

The second main component of this solution is the source file T2Crypto. This is a class (T2Crypto) that encapsulates the necessary logic to implement this solution.

Most implementations of this solution will also use T2’s version of the SQLCipher library.



Figure Lost Password Scheme Components

# Key Handling Overview



Figure 2 Key Handling Overview

## Initialization

* (A) Random Intermediate Key (RIKey) is created and database is initialized to use this key.
* (B) Locking Key is derived from the PIN using KDF algorithm. Master Key is created by encrypting the RIKey with the Locking Key. Master Key is saved to device storage.
* (D) Secondary Locking key is derived from the Security question answers using KDF algorithm. Backup Key is created by encrypting the RIKey with the Locking Key. Backup Key is saved to device storage.

## Login/Data Access With Pin

* (F) Locking Key is derived from the PIN.
* (G) RIKey is re-constructed by decrypting the Master Key using the Locking Key.
* If the RIKey matches the key that the database was initialized with then Login/Data access is successful.

## Login/Data Access with Security Answers

* (H) Secondary Locking Key is derived from the Security Question Answers.
* (G) RIKey is re-constructed by decrypting the Backup Key using the Secondary Locking Key.
* If the RIKey matches the key that the database was initialized with then Login/Data access is successful.

# Native Client Interaction with T2Crypto

The following is a description of the necessary client interaction with the T2Crypto class.

**See Demo IOS App at:git.t2.local:/git/mobile/ios/FipsPasswordTest.git**

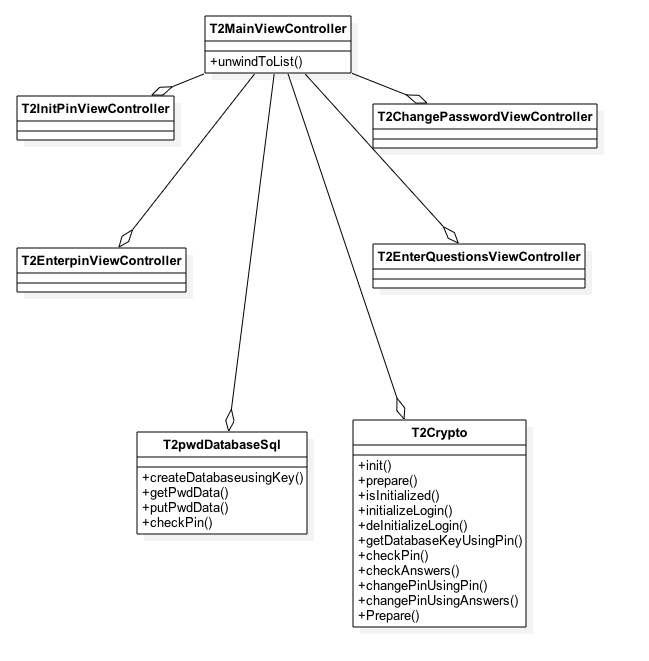


Figure Demo App Class Diagram

User of T2Crypto engine (T2MainViewController) (no error handling shown)

Responsibilities

***On initialization:***

Initialize T2Crypto engine with PIN, and security question answers

Call T2Crypto engines getDatabseKeyUsingPin function to get database key

Initialize sqlCipher using database key

Create database tables

ex:

int result = [t2Crypto initializeLogin:jsonPinAndAnswers];

if (result == T2Success)

NSString \*databaseKey = [t2Crypto  getDatabseKeyUsingPin :\_enteredPIN];

           mySQL.databaseKey = databaseKey;

           [mySQL createDbTables];

***On Login attempt:***

Check that pin is correct

Call T2Crypto engines getDatabseKeyUsingPin function to get database key

Initialize sqlCipher using database key

  ex:

  int result = [t2Crypto checkPin:\_enteredPIN];

 if (result == T2Success)

NSString \*databaseKey = [t2Crypto  getDatabseKeyUsingPin :\_enteredPIN];

           mySQL.databaseKey = databaseKey;

***On Login via answers attempt:***

             Check answers

Call T2Crypto engines getDatabseKeyUsingPin function to get database key

Initialize sqlCipher using database key

 ex:

 int result = [t2Crypto checkAnswers:\_iniPinDialogData];

 if (result == T2Success) {

[T2Crypto changePinUsingAnswers:\_iniPinDialogData]

     NSString \*databaseKey = [t2Crypto  getDatabseKeyUsingPin :\_enteredPIN];

           mySQL.databaseKey = databaseKey;

***On change password attempt:***

             Check pin

Call T2Crypto engines getDatabseKeyUsingPin function to get database key

Initialize sqlCipher using database key

 ex:

 int result = [t2Crypto checkPin:previousPin];

 if (result == T2Success) {

[T2Crypto changePinUsingAnswers:previousPin :newPin]

     NSString \*databaseKey = [t2Crypto  getDatabseKeyUsingPin :\_enteredPIN];

           mySQL.databaseKey = databaseKey;

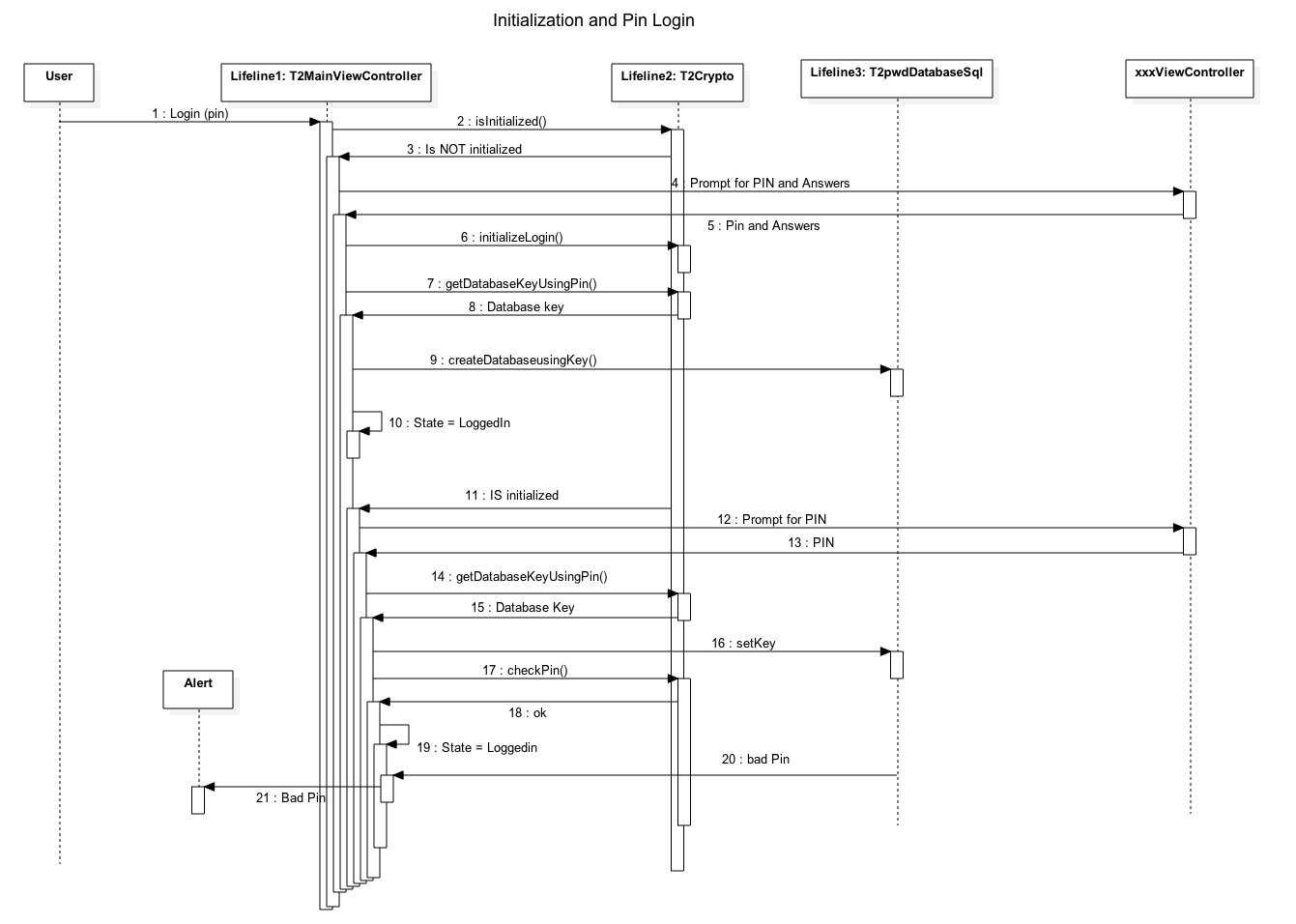


Figure 4 Initialization and Login with Pin

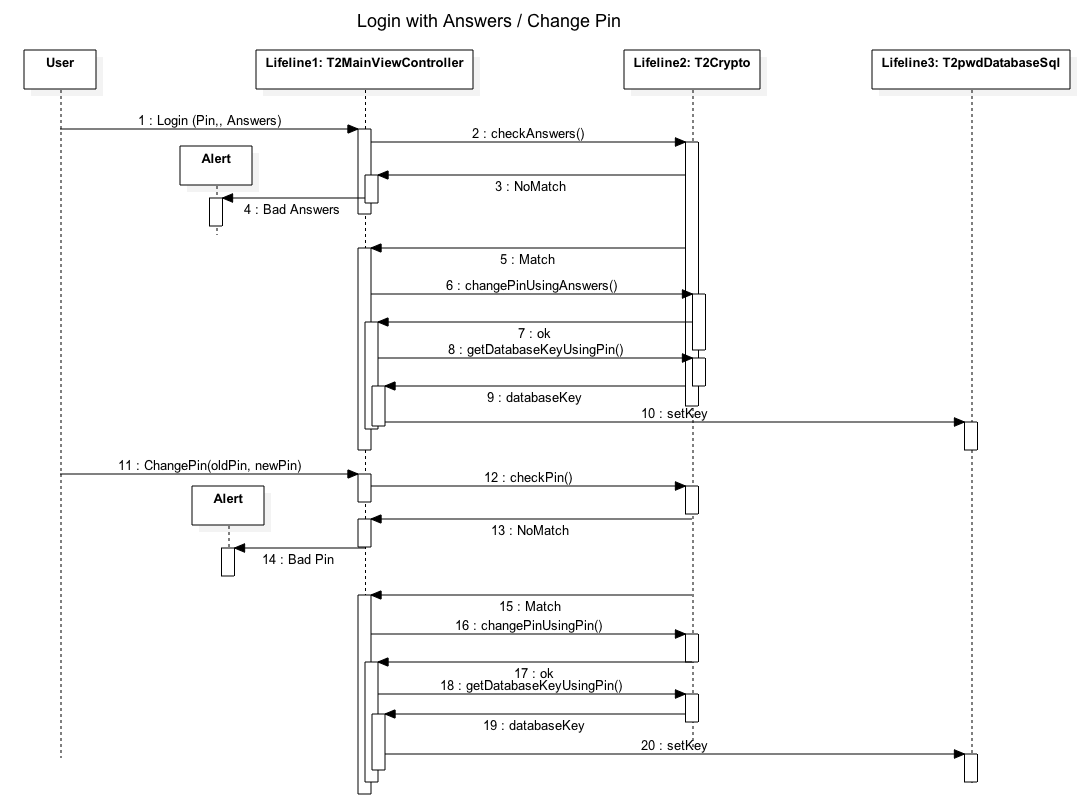


Figure 5 Login with Answers and Change Pin

# Cordova T2Crypto Module

## Introduction

Javascript interface file for Cordova plugin module t2crypto

**See git.t2.local:/git/mobile/cross-platform/cordova-plugin-trcrypto.git**

## Globals

[T2Crypto.prototype.changePinUsingAnswers](file:///C:\Users\scott.coleman\Desktop\index.html#//apple_ref/js/data/T2Crypto.prototype.changePinUsingAnswers)

Changes pin - checks answers to make sure they are correct

[T2Crypto.prototype.changePinUsingPin](file:///C:\Users\scott.coleman\Desktop\index.html#//apple_ref/js/data/T2Crypto.prototype.changePinUsingPin)

Changes pin - checks previous pin to make sure they are correct

[T2Crypto.prototype.checkAnswers](file:///C:\Users\scott.coleman\Desktop\index.html#//apple_ref/js/data/T2Crypto.prototype.checkAnswers)

Checks supplied answers to see if they are the correct answers for the database

[T2Crypto.prototype.checkPin](file:///C:\Users\scott.coleman\Desktop\index.html#//apple_ref/js/data/T2Crypto.prototype.checkPin)

Checks supplied pin to see if it's the correct key for the database

[T2Crypto.prototype.deinitializeLogin](file:///C:\Users\scott.coleman\Desktop\index.html#//apple_ref/js/data/T2Crypto.prototype.deinitializeLogin)

De-initializes Crypto - Clears all saved keys

[T2Crypto.prototype.getDatabaseKeyUsingPin](file:///C:\Users\scott.coleman\Desktop\index.html#//apple_ref/js/data/T2Crypto.prototype.getDatabaseKeyUsingPin)

Gets the database key based on a PIN

[T2Crypto.prototype.initializeLogin](file:///C:\Users\scott.coleman\Desktop\index.html#//apple_ref/js/data/T2Crypto.prototype.initializeLogin)

Initializes login - Sets up database, Sets up saved keys

[T2Crypto.prototype.initT2Crypto](file:///C:\Users\scott.coleman\Desktop\index.html#//apple_ref/js/data/T2Crypto.prototype.initT2Crypto)

Initializes T2Crypto library, creates RIKey

[T2Crypto.prototype.isInitialized](file:///C:\Users\scott.coleman\Desktop\index.html#//apple_ref/js/data/T2Crypto.prototype.isInitialized)

Checks to see if the database has been initialized with pin and answers

### T2Crypto.prototype.changePinUsingAnswers

Changes pin - checks answers to make sure they are correct

T2Crypto.prototype.changePinUsingAnswers = function(

loginString,

successCallback,

errorCallback) {

argscheck.checkArgs(

'sfF',

'T2Crypto.changePinUsingAnswers',

arguments);

exec(

successCallback,

errorCallback,

"T2Crypto",

"changePinUsingAnswers",

[loginString]);

};

##### Fields

loginString

formatted pin and database answers

successCallback

Function to call on success

errorCallback

(OPTIONAL) Function to call on error

##### Return Value

OnError: Error Code

##### Discussion

JSON Keys: KEY\_PIN KEY\_SECURITY\_ANSWER\_1 KEY\_SECURITY\_ANSWER\_2 KEY\_SECURITY\_ANSWER\_3

### T2Crypto.prototype.changePinUsingPin

Changes pin - checks previous pin to make sure they are correct

T2Crypto.prototype.changePinUsingPin = function(

pin,

successCallback,

errorCallback) {

argscheck.checkArgs(

'sfF',

'T2Crypto.changePinUsingPin',

arguments);

exec(

successCallback,

errorCallback,

"T2Crypto",

"changePinUsingPin",

[pin]);

};

##### Fields

pin

Previous (existing) PIN

successCallback

Function to call on success

errorCallback

(OPTIONAL) Function to call on error

##### Return Value

OnSuccess: N/A, OnError: Error Code

### T2Crypto.prototype.checkAnswers

Checks supplied answers to see if they are the correct answers for the database

T2Crypto.prototype.checkAnswers = function(

answers,

successCallback,

errorCallback) {

argscheck.checkArgs(

'sfF',

'T2Crypto.checkAnswers',

arguments);

exec(

successCallback,

errorCallback,

"T2Crypto",

"checkAnswers",

[answers]);

};

##### Fields

answers

Formatted database answers

successCallback

Function to call on success

errorCallback

(OPTIONAL) Function to call on error

##### Return Value

OnSuccess: 1 if correct answers, 0 if not, OnError: Error Code

##### Discussion

JSON Keys: KEY\_SECURITY\_ANSWER\_1 KEY\_SECURITY\_ANSWER\_2 KEY\_SECURITY\_ANSWER\_3

### T2Crypto.prototype.checkPin

Checks supplied pin to see if it's the correct key for the database

T2Crypto.prototype.checkPin = function(

pin,

successCallback,

errorCallback) {

argscheck.checkArgs(

'sfF',

'T2Crypto.checkPin',

arguments);

exec(

successCallback,

errorCallback,

"T2Crypto",

"checkPin",

[pin]);

};

##### Fields

pin

PIN to use

successCallback

Function to call on success

errorCallback

(OPTIONAL) Function to call on error

##### Return Value

OnSuccess: 1 if correct key, 0 if not, OnError: Error Code

### T2Crypto.prototype.deinitializeLogin

De-initializes Crypto - Clears all saved keys

T2Crypto.prototype.deinitializeLogin = function(

successCallback,

errorCallback) {

argscheck.checkArgs(

'fF',

'T2Crypto.deinitializeLogin',

arguments);

exec(

successCallback,

errorCallback,

"T2Crypto",

"deinitializeLogin",

[]);

};

##### Fields

successCallback

Function to call on success

errorCallback

(OPTIONAL) Function to call on error

##### Return Value

OnSuccess: N/A, OnError: Error Code

### T2Crypto.prototype.getDatabaseKeyUsingPin

Gets the database key based on a PIN

T2Crypto.prototype.getDatabaseKeyUsingPin = function(

pin,

successCallback,

errorCallback) {

argscheck.checkArgs(

'sfF',

'T2Crypto.getDatabaseKeyUsingPin',

arguments);

exec(

successCallback,

errorCallback,

"T2Crypto",

"getDatabaseKeyUsingPin",

[pin]);

};

##### Fields

pin

PIN to use

successCallback

Function to call on success

errorCallback

(OPTIONAL) Function to call on error

##### Return Value

OnSuccess: Database key, OnError: Error Code

### T2Crypto.prototype.initializeLogin

Initializes login - Sets up database, Sets up saved keys

T2Crypto.prototype.initializeLogin = function(

loginText,

successCallback,

errorCallback) {

// argscheck.checkArgs('fF', 'T2Crypto.initializeLogin', arguments);

exec(

successCallback,

errorCallback,

"T2Crypto",

"initializeLogin",

[loginText]);

};

##### Fields

loginText

Formatted login text

successCallback

Function to call on success

errorCallback

(OPTIONAL) Function to call on error

##### Return Value

OnSuccess: N/A, OnError: Error Code

##### Discussion

JSON Keys: KEY\_PIN KEY\_SECURITY\_ANSWER\_1 KEY\_SECURITY\_ANSWER\_2 KEY\_SECURITY\_ANSWER\_3

### T2Crypto.prototype.initT2Crypto

Initializes T2Crypto library, creates RIKey

T2Crypto.prototype.initT2Crypto = function(

successCallback,

errorCallback) {

argscheck.checkArgs(

'fF',

'T2Crypto.initT2Crypto',

arguments);

exec(

successCallback,

errorCallback,

"T2Crypto",

"initT2Crypto",

[]);

};

##### Fields

successCallback

Function to call on success

errorCallback

(OPTIONAL) Function to call on error

##### Return Value

OnSuccess: N/A, OnError: Error Code

### T2Crypto.prototype.isInitialized

Checks to see if the database has been initialized with pin and answers

T2Crypto.prototype.isInitialized = function(

successCallback,

errorCallback) {

argscheck.checkArgs(

'fF',

'T2Crypto.isInitialized',

arguments);

exec(

successCallback,

errorCallback,

"T2Crypto",

"isInitialized",

[]);

};

##### Fields

successCallback

Function to call on success

errorCallback

(OPTIONAL) Function to call on error

##### Return Value

OnSuccess: 1 if initialized, 0 if not , OnError: Error Code

# Cross Platform App Interaction With T2Crypto Cordova Module

**See git.t2.local:/git/mobile/crossplatform/IonicPasswordTestApp.git**

API Usage:

***On initialization:***

ex:

t2crypto.initT2Crypto;

t2crypto.initializeLogin(JSON login string, success, error);

var error = function(message) {

console.log("failed, error code: " + message)

};

  var success = function(pin) {

var db = window.sqlitePlugin.openDatabase({name: "myencrypt1.db", password: pin}, onOpenSuccess);

};

 t2crypto.getDatabaseKeyUsingPin(pin, success, error);

***On Login attempt:***

 t2crypto.checkPin(enteredPin, success, error);

***On Login via answers attempt:***

t2crypto.checkAnswers(formattedAnswers, success, error);

***On change password attempt:***

    t2crypto.changePinUsingAnswers(formattedAnswers, success, error);

    t2crypto.changePinUsingPin(previousPin, success, error);