Smart Dongle™ Implementation Guide

2/24/2009

Table of Contents

I. Example Source Code For Windows

a SmartDongle “C” Source Code

Examples: Change the keys from the demo keys given below in the example code to your unique key values:

P1 = 0xec6cc589aefd1e75;
P2 = 0xfcec0a6a82747b3f;

1. Example 1: Simple method to check for valid SmartDongle

```
/*
   example1.c: Simple method of securing software, just checks for SmartDongle
               presence.

   www.smartdongle.com

   MicroWorks, Inc.
   2808 North Cole Road
   Boise, ID 83704
*/

#include <stdio.h>
#include “smartdongle.h”
#include “sd_eeprom.h”


int __cdecl main(int argc, char* argv[])
{
    /* SmartDongle  Demo Keys */
    unsigned __int64 P1 = 0xec6cc589aefd1e75;
    unsigned __int64 P2 = 0xfcec0a6a82747b3f;
    int error;


    /* Unlock SmartDongle with 0 length read request */
    error = SmartDongleRead(P1, P2, 0, NULL, 0);
    if (error)
    {
        printf (“Failed to access a valid SmartDongle\n”);
    }
    else
    {
        printf(“A valid SmartDongle is present\n”);
    }

   return 0;
}
```

2. Example 2a: Write some application data to a SmartDongle

```
/*
   example2a.c:  Write some application data to a SmartDongle.


   www.smartdongle.com

   MicroWorks, Inc.
   2808 North Cole Road
   Boise, ID 83704
*/

#include <stdio.h>
#include “smartdongle.h”
#include “sd_eeprom.h”
#include “sd_uskerr.h”




int __cdecl main (int argc, char *argv[]) {

   /* SmartDongle demo keys. */
   unsigned __int64 P1 = 0xec6cc589aefd1e75;
   unsigned __int64 P2 = 0xfcec0a6a82747b3f;

   /*
      Your application data here.
   */
   unsigned char data[] = “This is data written to the SmartDongle, it can be serial
 numbers, program parameters, etc.  Note that this data can be read by observing activity
on the USB bus. Refer to example3a.c and example3b.c for example of encrypting your
application data.”
;
   char errorString[QTS_ERROR_STRING_LENGTH_MAX];
   int error;
   unsigned short address;



   /*
      Turn SmartDongle Red LED on to indicate write in progress.
   */
   SmartDongleLedRed();


   /*
```

```
    Write a user string to SmartDongle.
  */
  address = 0;
  error = SmartDongleWrite(P1, P2, address, data, strlen(data) + 1);
  if (error) {

    SmartDongleGetErrorString(errorString, error);
    printf("%s\n", errorString);
  } else {

    printf("Success: Wrote user application data to SmartDongle\n");
    SmartDongleLedGreen();
  }


  return 0;
}
```

3. Example 2b: Read the application data written in Example 2a

```
/*
  example2b.c: Read application data written to SmartDongle in example2a.c

  www.smartdongle.com

  MicroWorks, Inc.
  2808 North Cole Road
  Boise, ID 83704
*/

#include <stdio.h>
#include "smartdongle.h"
#include "sd_eeprom.h"
#include "sd_uskerr.h"

int __cdecl main (int argc, char *argv[]) {

  /* SmartDongle demo keys. */
  unsigned __int64 P1 = 0xec6cc589aefd1e75;
  unsigned __int64 P2 = 0xfcec0a6a82747b3f;

  /*
    Your application data here.
  */
  unsigned char data[] = "This is data written to the SmartDongle, it can be
 serial numbers, program parameters, etc.  Note that this data can be read by
observing activity on the USB bus. Refer to example3a.c and example3b.c for
 example of encrypting your application data."
;
  unsigned char *in;
```

```
    unsigned short address;
    int length;
    char errorString[QTS_ERROR_STRING_LENGTH_MAX];
    int error;



    /*
       Allocate input buffer for expected application data size
       plus one for NULL terminator.
    */
    length = strlen(data) + 1;
    in = (unsigned char *)malloc(length);
    if (in == NULL) {

       error = USK_ERR_MEM;
       SmartDongleGetErrorString(errorString, error);
       printf("%s\n", errorString);
       return 1;
    }


    /*
       Verify the applicaton data that was written in example2a.c

       This is just an exercise, a developer could do any number
       of things with the application data here.
    */
    address = 0;
    error = SmartDongleRead(P1, P2, address, in, length);
    if (error) {

       SmartDongleGetErrorString(errorString, error);
       printf("%s\n", errorString);
       return 1;
    } else {

       if (strcmp (in, data) == 0) {

          printf("Success: SmartDongle has application data\n");
       } else {

          printf ("Failed: Data does not match application data\n");
          return 1;
       }
    }

    return 0;
}
```

If you store data on a SmartDongle you may wish to encrypt the data transmitted over the USB bus.

Examples 3a and 3b below use AES encryption. A developer could use example3a.c as a utility to write passwords, serial numbers, etc to a SmartDongle that can be read back and used in an application as demonstrated by example 3b.c

4. Example 3a: Encrypt a user string using AES and write this string to a SmartDongle

```
/*
   example3a.c:  Example that encrypts a user string and
                 writes the encrypted string to SmartDongle.

   www.smartdongle.com

   MicroWorks, Inc.
   2808 North Cole Road
   Boise, ID 83704
*/

#include <stdio.h>
#include <malloc.h>
#include “smartdongle.h”
#include “sd_eeprom.h”
#include “sd_uskerr.h”
#include “aes.h”


int main(int argc, char* argv[])
{
   /* SmartDongle demo keys. */
   unsigned __int64 P1 = 0xec6cc589aefd1e75;
   unsigned __int64 P2 = 0xfcec0a6a82747b3f;

   /*
      This is a demo 128 bit encryption key,
      chose some another 16 character key for your project.
   */
   unsigned char AESkey[] = “7obdgEn2’TH>]SE|”;

   /*
      Demo secret
   */
   unsigned char plainText[] = “This is the secret written to SmartDongle,
it can be passwords, serial numbers, program parameters, etc”
;

   char errorString[QTS_ERROR_STRING_LENGTH_MAX];
```

```
SmartDongleCipher *cipher;
SmartDongleCipher cipherInput;
unsigned char *input;
int size, error;
unsigned short address;



/*
   Turn SmartDongle Red LED on to indicate write in progress.
*/
SmartDongleLedRed();


/*
   Encrypt your secret using AES with a block size of 128
*/
cipher =  SmartDongleEncode(
   plainText,
   strlen((char *)plainText) + 1,
   128,
   AESkey);
if (cipher == NULL) {

   error = USK_ERR_UNKNOWN;
   printf("Error encoding user secret: illegal AES block size?\n");
   return 1;
}

/*
   Write your encrypted secret to SmartDongle memory.
*/
address = 0;
size = (cipher->AESblockSize / 8) * cipher->numberOfBlocks;
error = SmartDongleWrite(P1, P2, address, cipher->data, size);
if (error) {

   SmartDongleGetErrorString(errorString, error);
   printf("Write Failed %d, %s\n", error, errorString);
   free(cipher->data);
   free(cipher);
   return 1;
}


/*
   Allocate input for cipher read from SmartDongle
*/
input = (unsigned char *)malloc(size);
if (input == NULL) {
```

```
    error =  USK_ERR_MEM;
    SmartDongleGetErrorString(errorString, error);
    printf("Error %d, %s\n", error, errorString);
    free(cipher->data);
    free(cipher);
    return 1;
}


/*
    Read encrypted data back from SmartDongle
*/
address = 0;
error = SmartDongleRead(P1, P2, address, input, size);
if (error) {

    SmartDongleGetErrorString(errorString, error);
    printf("Read Failed %d, %s\n", error, errorString);
    free(cipher->data);
    free(cipher);
    free(input);
    return 1;
}


/*
    Decrypt
*/
cipherInput.AESblockSize = cipher->AESblockSize;
cipherInput.numberOfBlocks = cipher->numberOfBlocks;
cipherInput.data = input;
SmartDongleDecode(&cipherInput, AESkey);


/*
    Compare decrypted cipher to expected user data.
    Compare using orignal secret length since the decryped cipher may have
    been padded with zeroes.
*/
if (memcmp(cipherInput.data, plainText, strlen((char *)plainText)) == 0) {

    printf ("Success: Your encrypted secret was written to SmartDongle\n");

    /* Indicate success */
    SmartDongleLedGreen();
} else {

    printf ("Fail: Data read from SmartDongle did not match\n");
    free(cipher->data);
```

```
        free(cipher);
        free(input);
        return 1;
    }

    free(cipher->data);
    free(cipher);
    free(input);
    return 0;
}
```

5. Example 3b: Read and decrypt the secret written in Example 3a

```
/*
    example3b.c:  Example that read and decrypts a user string from
                a SmartDongle as written in example3a.c


    www.smartdongle.com

    MicroWorks, Inc.
    2808 North Cole Road
    Boise, ID 83704
*/

#include <stdio.h>
#include <malloc.h>
#include “smartdongle.h”
#include “sd_eeprom.h”
#include “sd_uskerr.h”
#include “aes.h”


int main(int argc, char* argv[])
{
    /* SmartDongle demo keys. */
    unsigned __int64 P1 = 0xec6cc589aefd1e75;
    unsigned __int64 P2 = 0xfcec0a6a82747b3f;

    /*
        This is a demo 128 bit encryption key,
        chose some another 16 character key for your project.
    */
    unsigned char AESkey[] = “7obdgEn2’TH>]SE|”;

    /*
        Demo secret
    */
    unsigned char plainText[] = “This is the secret written to SmartDongle,
 it can be passwords, serial numbers, program parameters, etc”
```

;

```
    char errorString[QTS_ERROR_STRING_LENGTH_MAX];
    SmartDongleCipher cipher;
    unsigned char *input;
    int plainTextLength;
    int size, error;
    unsigned short address;


    /*
       Calculate the number of AES blocks to be read from SmartDongle,
       rounding block count up to nearest block size.
    */
    plainTextLength = strlen((char *)plainText) + 1; // length + null character
    cipher.AESblockSize = 128;
    cipher.numberOfBlocks =
       (plainTextLength * 8) / cipher.AESblockSize +
       ((plainTextLength * 8 % cipher.AESblockSize > 0) ? 1 : 0);


    /*
       Allocate data string for cipher read from SmartDongle
       Note that AES block size is length in bits.
    */
    size = (cipher.AESblockSize / 8) * cipher.numberOfBlocks;
    input = (unsigned char *)malloc(size);
    if (input  == NULL) {

       error =  USK_ERR_MEM;
       SmartDongleGetErrorString(errorString, error);
       printf("Error %d, %s\n", error, errorString);
       return 1;
    }


    /*
       Read encrypted data from SmartDongle
    */
    address = 0;
    error = SmartDongleRead(P1, P2, address, input, size);
    if (error) {

       SmartDongleGetErrorString(errorString, error);
       printf("Read Failed %d, %s\n", error, errorString);
       free(input);
       return 1;
    }
```

```
    /*
       Decrypt
    */
    cipher.data = input;
    SmartDongleDecode(&cipher, AESkey);


    /*
       Compare to original secret, using orignal secret length.

       The data comparison is just an exercise, the developer can do
       any number of things with the decrypted user data here.
    */
    if (memcmp(input, plainText, strlen((char *)plainText)) == 0) {

       printf ("Success: Your encrypted secret was found on SmartDongle\n");
    } else {

       printf ("Fail: Your secret was not found on SmartDongle\n");
    }

    free(input);
    return 0;
}
```

6. Example 4: Read SmartDongle serial number

```
/*
   example4.c

   Retrieves serial number of a single SmartDongle on the host.


   www.smartdongle.com

   MicroWorks, Inc.
   2808 North Cole Road
   Boise, ID 83704
*/

#include <stdio.h>
#include "smartdongle.h"
#include "sd_cp2.h"

int __cdecl main(int argc, char* argv[])
{
   char sn[CP2_SERIALNUM_LENGTH+1];
   char errorString[QTS_ERROR_STRING_LENGTH_MAX];
   int error;
```

```
  error = SmartDongleGetSerialNumber(sn);
  if (error)
  {
    SmartDongleGetErrorString(errorString, error);
    printf("%s\n", errorString);
  }
  else
  {
    printf("Serial Number: %s\n", sn);
  }

  printf("\nPress any key to exit\n");
  getchar();

  return 0;
}
```

b. SmartDongle C# Example Source Code

Examples
Change the keys from the demo keys given below in the example code to your unique key values:

P1 = 0xec6cc589aefd1e75;
P2 = 0xfcec0a6a82747b3f;

1. Example 1: Simple method to check for valid SmartDongle

```
// Example1.cs
using System;
namespace Example1 {
    /// <summary>
    /// Simple program that verifies if a valid SmartDongle is
plugged in.
    ///
    /// MicroWorks, Inc.
    /// 2808 North Cole Road
    /// Boise, ID 83704
    ///
    /// http://www.smartdongle.com
    ///
    class Example1 {
        [STAThread]
        static void Main(string[] args) {
            UInt64 P1, P2;
            SmartDongle.SmartDnglError error;
            System.Text.ASCIIEncoding  encoding = new
System.Text.ASCIIEncoding();

            /* Your keys here */
            P1=0xec6cc589aefd1e75;
            P2=0xfcec0a6a82747b3f;
```

```
            /* Simply check for presence of valid SmartDongle.  */
                        error = SmartDongle.Read(P1, P2, 0, new byte[0], 0);
                        if (error != SmartDongle.SmartDnglError.UskOk) {

            System.Console.WriteLine(SmartDongle.GetErrorString(error));
                            // handle error
                        }
                        else {
                            System.Console.WriteLine("Success: Valid
            SmartDongle Found\n");
                        }
                        System.Console.WriteLine("Press <Enter> to Exit");
                        System.Console.ReadLine();
                    }
                }
            }
```

2. Example 2a: Write some application data to a SmartDongle

```
// Example2a.cs
using System;
using System.Text;
using System.IO;

namespace Example {
    ///  Write some misc application data to a SmartDongle.
    ///
    /// MicroWorks, Inc.
    /// 2808 North Cole Road
    /// Boise, ID 83704
    ///
    /// http://www.smartdongle.com
    ///
    class Example2a {

        [STAThread]

         Static void Main(string[] args) {
            UInt64 P1, P2;
            SmartDongle.SmartDnglError error;
            System.Text.ASCIIEncoding  encoding = new System.Text.ASCIIEncoding();

            /* Some application data */
            const string yourData = "This is data written to the SmartDongle, it can be serial
numbers, program parameters, etc.  Note that this data can be read by observing activity on
the USB bus. Refer to Example3a.cs and Example3b.cs for example of encrypting your application
data.";

            /* Your SmartDongle keys go here */
            P1 = 0xec6cc589aefd1e75;
            P2 = 0xfcec0a6a82747b3f;

            System.Console.WriteLine(sizeof(char));
```

```
            /*
             *  Write your data to SmartDongle.
             *  Note: The number of writes is limited, SmartDongle uses a Catalyst
             *  CAT25C256 SPI eprom, which has a life expectancy of approximately
             *  1 Million write cycles.  Reads are unlimited.
             */
             UInt16 address = 0;
             byte[] plainText = Encoding.UTF8.GetBytes(yourData);

             System.Console.WriteLine("Size of byte array: "+plainText.Length);

             error = SmartDongle.Write(P1, P2, address, plainText, plainText.Length);

             if (error == SmartDongle.SmartDnglError.UskOk) {
                 System.Console.WriteLine("Success: Wrote application data to
                                           SmartDongle\n");
             }
             else {
                 System.Console.WriteLine(SmartDongle.GetErrorString(error));
                 System.Console.WriteLine("Failed: Could not write application data to
                                           SmartDongle\n");
             }

             System.Console.WriteLine("Press <Enter> to Exit");
             System.Console.ReadLine();
         }
     }
}
```

3. Example 2b: Read the application data written in Example 2a

```
// Example2b.cs
using System;
using System.Security.Cryptography;
using System.Text;
using System.IO;

namespace Example
{
    /// Read application data written to SmartDongle in Example2a.cs
    ///
    /// MicroWorks, Inc.
    /// 2808 North Cole Road
    /// Boise, ID 83704
    ///
    /// http://www.smartdongle.com
```

```
class Example2b
 {
     [STAThread]

     static void Main(string[] args)
     {
         UInt64 P1, P2;
         UInt16 address = 0;

         SmartDongle.SmartDnglError error;

         System.Text.ASCIIEncoding encoding = new
               System.Text.ASCIIEncoding();

         /* This is the application data you are looking for from Example 2A
         */
         const string yourData = "This is data written to the SmartDongle, it
               can be serial numbers, program parameters, etc.  Note that
               this data can be read by observing activity on the USB bus.
               Refer to Example3a.cs and Example3b.cs for example of
               encrypting your application data.";

         /* Your SmartDongle keys here */
         P1 = 0xec6cc589aefd1e75;
         P2 = 0xfcec0a6a82747b3f;

         /*
          *  Read a byte array from the SmartDongle with the expected length
               of your data
          */
         byte[] datum = Encoding.UTF8.GetBytes(yourData);
         byte[] inBuffer = new byte[yourData.Length];

         error = SmartDongle.Read(P1, P2, address, inBuffer,
                                  yourData.Length);

         if (error != SmartDongle.SmartDnglError.UskOk)
         {
             System.Console.WriteLine(SmartDongle.GetErrorString(error));
             return;
         }

        /*
          *  As an exercise, simply comparing data read from SmartDongle to
          *  the expected user data.  A developer could do any number of
          *  things
          *  with the data read here.
          */
```

```
            int i;
            for (i = 0; (i < datum.Length) && (datum[i] == inBuffer[i]); i++) ;
            if (i == datum.Length)
            {
                System.Console.WriteLine("Success: Found Valid Application
Data\n");
            }
            else
            {
                System.Console.WriteLine("Failed: Invalid Application Data\n");
            }

            System.Console.WriteLine("Press <Enter> to Exit");
            System.Console.ReadLine();
        }
    }
}
```

If you store data on a SmartDongle you may wish to encrypt the data transmitted over the USB bus.

Examples 3a and 3b below use AES encryption. A developer could use example3a.c as a utility to write passwords, serial numbers, etc to a SmartDongle that can be read back and used in an application as demonstrated by example3b.c

4. Example 3a: Encrypt a user string using AES and write this string to a SmartDongle

```
// Example3a.cs
using System;
using System.Security.Cryptography;
using System.Text;
using System.IO;
namespace Example {
    /// Write then verify an encrypted secret on a SmartDongle.
    ///
    /// MicroWorks, Inc.
    /// 2808 North Cole Road
    /// Boise, ID 83704
    ///
    /// http://www.smartdongle.com
    ///
    class Example3a {
        [STAThread]
        static void Main(string[] args) {
            try {
                UInt64 P1, P2;
                SmartDongle.SmartDnglError error;
                System.Text.ASCIIEncoding  encoding = new
System.Text.ASCIIEncoding();
                /*
                 *  Padding is zeroes and CipherMode has been set to
                 *  Electronic Codebook (ECB) to match the algorithm in the C
                    examples.
```

```
 *  The AES key and user secret below are for example only, the user
 *  should create a new key and secret for their application.
 *
 *  A System.Security.Cryptography.CryptographicException will
 *  occur if illegal AES key length or AES key size are given below.
*/
const string aes_key = "7obdgEn2'TH>]SE|";
const int aes_KeySize = 128;
const int aes_BlockSize = 128;
const string your_secret =  "This is the secret written to SmartDongle, it
  can be passwords, serial numbers, program parameters, etc";

Rijndael RijndaelAlg = Rijndael.Create();
RijndaelAlg.KeySize = aes_KeySize;
RijndaelAlg.BlockSize = aes_BlockSize;
RijndaelAlg.Mode = CipherMode.ECB;
RijndaelAlg.Padding = PaddingMode.Zeros;

/* Initial vector (rgbIV) is the same as the key as in the C example code
  */
byte[] rgbKey = encoding.GetBytes(aes_key);
byte[] rgbIV = rgbKey;

ICryptoTransform encrypt = RijndaelAlg.CreateEncryptor(rgbKey, rgbIV);
ICryptoTransform decrypt = RijndaelAlg.CreateDecryptor(rgbKey, rgbIV);

/* Convert secret into byte array, then encrypt */
byte[] plainText = Encoding.UTF8.GetBytes(your_secret);
byte[] cipherText = encrypt.TransformFinalBlock(plainText, 0,
                    plainText.Length);

/* Your SmartDongle keys go here */
P1 = 0xec6cc589aefd1e75;
P2 = 0xfcec0a6a82747b3f;

/* Write the encrypted secret to the begining of SmartDongle user memory.
*  Note: The number of writes is limited, SmartDongle uses a Catalyst
*  CAT25C256 SPI eprom, which has a life expectancy of approximately
*  1 Million write cycles.  Reads are unlimited.
*/
UInt16 address = 0;
error = SmartDongle.Write(
    P1, P2, address, cipherText, cipherText.Length);
if (error != SmartDongle.SmartDnglError.UskOk) {
    System.Console.WriteLine(SmartDongle.GetErrorString(error));
    // handle error
    return;
}

/*
 *  Do a data integrity check.
 *  Read the encrypted secret from the SmartDongle,
 *  decrypt and compare with original secret.
 */
byte[] input = new byte[cipherText.Length];
```

```
            error = SmartDongle.Read(P1, P2, address, input, cipherText.Length);
            if (error != SmartDongle.SmartDnglError.UskOk) {
                System.Console.WriteLine(SmartDongle.GetErrorString(error));
                // handle error
                return;
            }

            /* Decrypt */
            byte[] result = decrypt.TransformFinalBlock(input, 0, cipherText.Length);

            /* Compare to original secret, using orignal secret length */
            int i;
            for (i = 0; (i < plainText.Length) && (result[i] == plainText[i]); i++);
            if (i == plainText.Length) {
                System.Console.WriteLine("Success: Wrote Secret to SmartDongle\n");
            }
            else {
                System.Console.WriteLine("Failed: Writing Secret to SmartDongle\n");
            }
        }
        catch (Exception e) {
            Console.WriteLine("Error: {0}", e.Message);
        }
        System.Console.WriteLine("Press <Enter> to Exit");
        System.Console.ReadLine();
    }
  }
}
```

5. Example 3b: Read and decrypt the secret written in Example 3a

```
// Example3b.cs

using System;
using System.Security.Cryptography;
using System.Text;
using System.IO;

namespace Example
{
    /// <summary>
    /// Verify a secret on a SmartDongle.
    ///
    ///
    /// MicroWorks, Inc.
    /// 2808 North Cole Road
    /// Boise, ID 83704
    ///
    /// http://www.smartdongle.com
    ///
    /// </summary>
```

```
class Example3b
{
    [STAThread]
    static void Main(string[] args)
    {
        try
        {
            UInt64 P1, P2;
            SmartDongle.SmartDnglError error;
            System.Text.ASCIIEncoding  encoding = new System.Text.ASCIIEncoding();

            /*  Padding is zeroes and CipherMode has been set to
             *  Electronic Codebook (ECB) to match the algorithm in the C example.
             *
             *  The AES key and user secret below are for example only, the user
             *  should create a new key and secret for their application.
             *
             *  A System.Security.Cryptography.CryptographicException will
             *  occur if illegal AES key length or AES key size are given below.
            */
            const string aes_key = "7obdgEn2'TH>]SE|";
            const int aes_KeySize = 128;   // Size in bits
            const int aes_BlockSize = 128; // Size in bits
            const string your_secret = "This is the secret written to SmartDongle, it
                   can be passwords, serial numbers, program parameters, etc";

            Rijndael RijndaelAlg = Rijndael.Create();
            RijndaelAlg.KeySize = aes_KeySize;
            RijndaelAlg.BlockSize = aes_BlockSize;
            RijndaelAlg.Mode = CipherMode.ECB;
            RijndaelAlg.Padding = PaddingMode.None;

            byte[] rgbKey = encoding.GetBytes(aes_key);
            byte[] rgbIV = rgbKey;

            ICryptoTransform decrypt = RijndaelAlg.CreateDecryptor(rgbKey, rgbIV);

            /* Your SmartDongle keys go here */
            P1 = 0xec6cc589aefd1e75;
            P2 = 0xfcec0a6a82747b3f;

            /*
             *  Read the encrypted secret from the SmartDongle,
             *  decrypt and compare with original secret.
             */
            byte[] secret = Encoding.UTF8.GetBytes(your_secret);

            /* Cipher length is a multiple of AES blocksize.
              * Note that AES blocksize is in bits.
             */
            int cipherBlocks = secret.Length / (aes_BlockSize / 8);
            if (secret.Length % (aes_BlockSize / 8) > 0) {
                cipherBlocks++;
            }
```

```
                byte[] cipherText = new byte[cipherBlocks * (aes_BlockSize / 8)];
                UInt16 address = 0;
                error = SmartDongle.Read(P1, P2, address, cipherText,
                        cipherText.Length);
                if (error != SmartDongle.SmartDnglError.UskOk) {

                    System.Console.WriteLine(SmartDongle.GetErrorString(error));

                    // handle error
                    return;
                }

                /* Decrypt */
                byte[] result = decrypt.TransformFinalBlock(cipherText, 0,
                        cipherText.Length);

                /* Compare to original secret, using expected orignal secret length
                  */
                int i;
                for (i = 0; (i < secret.Length) && (result[i] == secret[i]); i++) ;
                if (i == secret.Length)
                {
                    System.Console.WriteLine("Success: Found Valid Secret\n");
                }
                else
                {
                    System.Console.WriteLine("Failed: Invalid Secret\n");
                }
            }
            catch (Exception e)
            {
                Console.WriteLine("Error: {0}", e.Message);
            }
            System.Console.WriteLine("Press <Enter> to Exit");
            System.Console.ReadLine();
        }
    }
}
```

6. Example 4: Read SmartDongle serial number

```
// Example4.cs
using System;
namespace Example4 {
    /// This will read the serial number of a single SmartDongle.
    ///
    /// MicroWorks, Inc.
    /// 2808 North Cole Road
    /// Boise, ID 83704
    ///
    /// http://www.smartdongle.com
    class Example4 {
        [STAThread]
        static void Main(string[] args) {
            SmartDongle.SmartDnglError error;
            System.Text.ASCIIEncoding  encoding = new System.Text.ASCIIEncoding();

            /* Get SmartDongle serial number byte array */
            string serialNumber = new string('0', SmartDongle.CP2_SERIALNUM_LENGTH);
            error = SmartDongle.GetSerialNumber(ref serialNumber);
            if (error != SmartDongle.SmartDnglError.UskOk) {
                System.Console.WriteLine(SmartDongle.GetErrorString(error));
                // handle error
            }
            else {
                System.Console.WriteLine("Serial Number: {0:G}\n", serialNumber);
            }
            System.Console.WriteLine("Press <Enter> to Exit");
            System.Console.ReadLine();
        }
    }
}
```

c. SmartDongle Visual Basic .NET Example Source Code

Examples
Edit the example code to use your unique key values, for example:

P1 = &HEC6CC589AEFD1E75
P2 = &HFCEC0A6A82747B3F

a. Example 1: Simple method to check for valid SmartDongle

‘ Example 1: Simple program that verifies if a valid SmartDongle is plugged in.
‘
‘ MicroWorks, Inc.
‘ 2808 North Cole Road

```
‘ Boise, ID 83704
‘
‘ http://www.smartdongle.com

Imports System

Module Example1

   Sub Main()

      Dim dongle As New SmartDongle()
      Dim P1 As Int64, P2 As Int64
      Dim dnglError As SmartDongle.SmartDnglError
      Dim buf() As Byte = New Byte(0) {}


      ‘
      ‘  Your keys here
      ‘
      P1 = &HEC6CC589AEFD1E75
      P2 = &HFCEC0A6A82747B3F

      ‘ Check if valid SmartDongle is present.
      ‘
      dnglError = dongle.Read(P1, P2, 0, buf, 0)
      If dnglError <> SmartDongle.SmartDnglError.UskOk Then
         System.Console.WriteLine(SmartDongle.GetErrorString(dnglError))
         ‘ handle error
      Else
         System.Console.WriteLine(“Success: Valid SmartDongle found”)
      End If

      System.Console.WriteLine(“”)
      System.Console.WriteLine(“Press <Enter> to Exit”)
      System.Console.ReadLine()

   End Sub

End Module
```

2. Example 2a: Write some application data to a SmartDongle

```
‘ Example 2a:   Write some misc application data to a SmartDongle.
‘
‘ MicroWorks, Inc.
‘ 2808 North Cole Road
‘ Boise, ID 83704
‘
‘ http://www.smartdongle.com
```

```
Imports System

Module Example2a

   Sub Main()

      Dim dongle As New SmartDongle()
      Dim P1 As Int64, P2 As Int64
      Dim dnglError As SmartDongle.SmartDnglError
      Dim address As Int16
      Dim buffer() As Byte
      Dim str As String = New String(c:=Convert.ToChar(&H0), count:=32)
      Dim yourData As String
      Dim encoding As System.Text.ASCIIEncoding = New System.Text.ASCIIEncoding()


      ‘ Some application data
      yourData = “This is data written to the SmartDongle, it can be serial numbers,
program parameters, etc.  Note that this data can be read by observing activity on the
USB bus. Refer to Example3a.vb and Example3b.vb for example of encrypting your
application data.”

      ‘  Your keys here
      P1 = &HEC6CC589AEFD1E75
      P2 = &HFCEC0A6A82747B3F

      ‘ Write a Byte array at begining of SmartDongle user memory.
      ‘
      ‘ Note: The number of writes is limited, SmartDongle uses a Catalyst
      ‘       CAT25C256 SPI eprom, which has a life expectancy of approximately
      ‘       1 Million writes.  Unlimited reads, of course.
      ‘
      buffer = New Byte(yourData.Length) {}
      buffer = encoding.GetBytes(yourData)
      address = 0 ‘ User memory address. Byte addressable from 0 to 32431
      dnglError = dongle.Write(P1, P2, address, buffer, buffer.Length)
      If dnglError = SmartDongle.SmartDnglError.UskOk Then

         System.Console.WriteLine(“Success: Wrote application data to SmartDongle”)
      Else

         System.Console.WriteLine(“Failed: Could not write application data to
SmartDongle”)
         System.Console.WriteLine(SmartDongle.GetErrorString(dnglError))

         ‘ handle error
      End If

      System.Console.WriteLine(“”)
      System.Console.WriteLine(“Press <Enter> to Exit”)
```

```
        System.Console.ReadLine()
    End Sub

End Module
```

3. Example 2b: Read the application data written in Example 2a

```
‘ Example 2b: Verify a secret stored on a SmartDongle.
‘
‘ MicroWorks, Inc.
‘ 2808 North Cole Road
‘ Boise, ID 83704
‘
‘ http://www.smartdongle.com

Imports System

Module Example2b

    Sub Main()

        Dim dongle As New SmartDongle()
        Dim P1 As Int64, P2 As Int64
        Dim dnglError As SmartDongle.SmartDnglError
        Dim address As Int16
        Dim buffer() As Byte
        Dim str As String = New String(c:=Convert.ToChar(&H0), count:=32)
        Dim yourData As String
        Dim yourDataBytes() As Byte
        Dim encoding As System.Text.ASCIIEncoding = New System.Text.ASCIIEncoding()
        Dim i As Integer


        ‘ This is the application data you are looking for from Example 2A
        yourData = “This is data written to the SmartDongle, it can be serial numbers,
program parameters, etc.  Note that this data can be read by observing activity on
the USB bus. Refer to Example3a.vb and Example3b.vb for example of encrypting
 your application data.”

        ‘  Your keys here
        P1 = &HEC6CC589AEFD1E75
        P2 = &HFCEC0A6A82747B3F

        ‘ Read SmartDongle user memory into a Byte array.
        buffer = New Byte(yourData.Length) {}
        address = 0 ‘ User memory address. Byte addressable from 0 to 32431
        dnglError = dongle.Read(P1, P2, address, buffer, yourData.Length)
        If dnglError <> SmartDongle.SmartDnglError.UskOk Then
            System.Console.WriteLine(SmartDongle.GetErrorString(dnglError))
            ‘ handle error
```

```
        End If

        yourDataBytes = encoding.GetBytes(yourData)

        ‘ As an exercise, simply comparing data read from SmartDongle to
        ‘ the expected user data.  A developer could do any number of
        ‘ tricks with the data here.
        For i = 1 To yourDataBytes.Length - 1
           If (buffer(i) <> yourDataBytes(i)) Then
              System.Console.WriteLine(“Failed: Invalid Application Data”)
              Exit For
           End If
        Next

        If i = yourDataBytes.Length Then
           System.Console.WriteLine(“Success: Found Valid Application Data”)
        End If

        System.Console.WriteLine(“”)
        System.Console.WriteLine(“Press <Enter> to Exit”)
        System.Console.ReadLine()

    End Sub

End Module
```

If you store data on a SmartDongle you may wish to encrypt the data transmitted over the USB bus.

Examples 3a and 3b below use AES encryption. A developer could use Example 3a as a utility to write passwords, serial numbers, etc to a SmartDongle that can be read back and used in an application as demonstrated by Example 3b

4. Example 3a: Encrypt a user string using AES and write this string to a SmartDongle

```
‘ Example 3a:  Write then verify an encrypted secret on a SmartDongle
‘
‘ MicroWorks, Inc.
‘ 2808 North Cole Road
‘ Boise, ID 83704
‘
‘ http://www.smartdongle.com

Imports System
Imports System.Security.Cryptography
Imports System.Text
Imports System.IO

Module Example2a
```

```
Sub Main()

    Dim dongle As New SmartDongle()
    Dim dnglError As SmartDongle.SmartDnglError


    ‘  Your keys here
    Dim P1 As Int64 = &HEC6CC589AEFD1E75
    Dim P2 As Int64 = &HFCEC0A6A82747B3F


    ‘ Your secret data
    Dim your_secret As String = “This is the secret written to SmartDongle,
it can be passwords, serial numbers, program parameters, etc”


    ‘ Padding is zeroes and CipherMode has been set to
    ‘ Electronic Codebook (ECB) to match the algorithm in the C examples.
    ‘
    ‘ The AES key and user secret below are for example only, the user
    ‘ should create a new key and secret for their application.
    ‘
    ‘  A System.Security.Cryptography.CryptographicException will
    ‘  occur if illegal AES key length or AES key size are given below.

    Dim aes_key As String = “7obdgEn2’TH>]SE|”
    Dim aes_KeySize As Integer = 128
    Dim aes_BlockSize As Integer = 128
    Dim RijndaelAlg As Rijndael
    Dim rgbKey() As Byte
    Dim rgbIV() As Byte
    Dim encrypt As ICryptoTransform
    Dim decrypt As ICryptoTransform
    Dim plainText() As Byte
    Dim cipherText() As Byte
    Dim result() As Byte

Try
    RijndaelAlg = Rijndael.Create()
    RijndaelAlg.KeySize = aes_KeySize
    RijndaelAlg.BlockSize = aes_BlockSize
    RijndaelAlg.Mode = CipherMode.ECB
    RijndaelAlg.Padding = PaddingMode.Zeros

    ‘ Initial vector (rgbIV) is the same as the key as in the C example code
    Dim encoding As System.Text.ASCIIEncoding = New System.Text.ASCIIEncoding()
    rgbKey = encoding.GetBytes(aes_key)
    rgbIV = rgbKey
```

```
        encrypt = RijndaelAlg.CreateEncryptor(rgbKey, rgbIV)
        decrypt = RijndaelAlg.CreateDecryptor(rgbKey, rgbIV)

        ‘ Convert secret into byte array, then encrypt
        plainText = encoding.GetBytes(your_secret)
        cipherText = encrypt.TransformFinalBlock(plainText, 0, plainText.Length)

    Catch ex As System.Security.Cryptography.CryptographicException
        MsgBox(“Failed to initialized Rijndael Encryption / Decryption Objects”)

        ‘ handle error
        Return
    End Try

        ‘ Write the encrypted secret to the begining of SmartDongle user memory.
        ‘
        ‘ Note: The number of writes is limited, SmartDongle uses a Catalyst
        ‘       CAT25C256 SPI eprom, which has a life expectancy of approximately
        ‘       1 Million writes.  Unlimited reads, of course.
        ‘

        Dim address As Int16 = 0 ‘ User memory address. Byte addressable from 0 to 32431
        dnglError = dongle.Write(P1, P2, address, cipherText, cipherText.Length)
        If dnglError <> SmartDongle.SmartDnglError.UskOk Then

            System.Console.WriteLine(“Failed: Writing Secret to SmartDongle”)
            System.Console.WriteLine(SmartDongle.GetErrorString(dnglError))

            ‘ handle error
            Return
        End If


        ‘ Do a data integrity check.
        ‘ Read the encrypted secret from the SmartDongle,
        ‘ decrypt and compare with original secret.
        Dim input(cipherText.Length - 1) As Byte
        dnglError = dongle.Read(P1, P2, address, input, input.Length)

        If dnglError <> SmartDongle.SmartDnglError.UskOk Then

            System.Console.WriteLine(“Failed: Reading Secret back from SmartDongle”)
            System.Console.WriteLine(SmartDongle.GetErrorString(dnglError))

            ‘ handle error
            Return
        End If


    Try
```

```
        ‘ Decrypt
        result = decrypt.TransformFinalBlock(input, 0, input.Length)

Catch ex As System.Security.Cryptography.CryptographicException

        MsgBox(“Failure of Rijndael decryption occurred”)

        ‘ handle error
        Return
End Try

        ‘ Compare to original secret, using orignal secret length since the
        ‘ result will be a multiple of block length and padded with zeroes
        Dim i As Integer
        For i = 1 To plainText.Length - 1
            If (result(i) <> plainText(i)) Then
                Exit For
            End If
        Next

        If i = plainText.Length Then
            System.Console.WriteLine(“Success: Wrote Secret to SmartDongle”)
        Else
            System.Console.WriteLine(“Failed: Error writing secret, data integrity
check failed”)
        End If

        System.Console.WriteLine(“”)
        System.Console.WriteLine(“Press <Enter> to Exit”)
        System.Console.ReadLine()
    End Sub

End Module
```

5. Example 3b: Read and decrypt the secret written in Example 3a

```
‘ Example 3b:  Verify a secret on a SmartDongle.
‘
‘ MicroWorks, Inc.
‘ 2808 North Cole Road
‘ Boise, ID 83704
‘
‘ http://www.smartdongle.com

Imports System
Imports System.Security.Cryptography
Imports System.Text
Imports System.IO

Module Example3b
```

```
Sub Main()

    Dim dongle As New SmartDongle()
    Dim dnglError As SmartDongle.SmartDnglError
    Dim encoding As System.Text.ASCIIEncoding = New System.Text.ASCIIEncoding()


    ‘  Your keys here
    Dim P1 As Int64 = &HEC6CC589AEFD1E75
    Dim P2 As Int64 = &HFCEC0A6A82747B3F


    ‘ Your secret data, make a copy in a byte array
    Dim your_secret As String = “This is the secret written to SmartDongle,
it can be passwords, serial numbers, program parameters, etc”
    Dim plainText() As Byte = encoding.GetBytes(your_secret)


    ‘ Padding is zeroes and CipherMode has been set to
    ‘ Electronic Codebook (ECB) to match the algorithm in the C examples.
    ‘
    ‘ The AES key and user secret below are for example only, the user
    ‘ should create a new key and secret for their application.
    ‘
    ‘  A System.Security.Cryptography.CryptographicException will
    ‘  occur if illegal AES key length or AES key size are given below.

    Dim aes_key As String = “7obdgEn2’TH>]SE|”
    Dim aes_KeySize As Integer = 128
    Dim aes_BlockSize As Integer = 128
    Dim RijndaelAlg As Rijndael
    Dim rgbKey() As Byte
    Dim rgbIV() As Byte
    Dim decrypt As ICryptoTransform
    Dim result() As Byte

Try
    RijndaelAlg = Rijndael.Create()
    RijndaelAlg.KeySize = aes_KeySize
    RijndaelAlg.BlockSize = aes_BlockSize
    RijndaelAlg.Mode = CipherMode.ECB
    RijndaelAlg.Padding = PaddingMode.Zeros

    ‘ Initial vector (rgbIV) is the same as the key as in the C example code
    rgbKey = encoding.GetBytes(aes_key)
    rgbIV = rgbKey

    decrypt = RijndaelAlg.CreateDecryptor(rgbKey, rgbIV)
```

```
Catch ex As System.Security.Cryptography.CryptographicException
    MsgBox("Failed to initialized Rijndael Decryption Objects")

    ' handle error
    Return
End Try


    '
    ' Cipher length read from SmartDongle is a multiple of AES blocksize.
    ' Note that AES blocksize is in bits.
    '
    Dim cipherBlocks As Integer = CType(your_secret.Length / (aes_BlockSize / 8),
 Integer)
    If (your_secret.Length Mod (aes_BlockSize / 8) > 0) Then
       cipherBlocks = cipherBlocks + 1
    End If


    '
    ' Read the encrypted secret from the SmartDongle,
    '
    Dim cipherText(CType((cipherBlocks * (aes_BlockSize / 8)) - 1, Integer)) As Byte
    Dim address As Int16 = 0 ' User memory address. Byte addressable from 0 to 32431
    dnglError = dongle.Read(P1, P2, address, cipherText, cipherText.Length)

    If dnglError <> SmartDongle.SmartDnglError.UskOk Then

       System.Console.WriteLine("Failed: Reading Secret back from SmartDongle")
       System.Console.WriteLine(SmartDongle.GetErrorString(dnglError))

       ' handle error
       Return
    End If

Try
    '
    ' Decrypt
    '
    result = decrypt.TransformFinalBlock(cipherText, 0, cipherText.Length)

Catch ex As System.Security.Cryptography.CryptographicException

    MsgBox("Failure of Rijndael decryption occurred")

    ' handle error
    Return
End Try

    ' Compare to original secret, using orignal secret length since the
```

```
        ‘ result will be a multiple of block length and padded with zeroes
        ‘ The data comparison here is just an exercise, the developer can do
        ‘ any number of things with the decrypted user data here.
        Dim i As Integer
        For i = 1 To plainText.Length - 1
            If (result(i) <> plainText(i)) Then
                Exit For
            End If
        Next

        If i = plainText.Length Then
            System.Console.WriteLine(“Success: Found Valid Secret”)
        Else
            System.Console.WriteLine(“Failed: Invalid Secret”)
        End If

        System.Console.WriteLine(“”)
        System.Console.WriteLine(“Press <Enter> to Exit”)
        System.Console.ReadLine()
    End Sub

End Module
```

6. Example 4: Read a SmartDongle Serial Number

```
‘ Example 4: Read a SmartDongle Serial Number
‘
‘ MicroWorks, Inc.
‘ 2808 North Cole Road
‘ Boise, ID 83704
‘
‘ http://www.smartdongle.com

Imports System

Module Example4

    Sub Main()

        Dim dongle As New SmartDongle()
        Dim dnglError As SmartDongle.SmartDnglError
        Dim serialNumber(SmartDongle.CP2_SERIALNUM_LENGTH) As Char


        dnglError = dongle.GetSerialNumber(serialNumber)
        Dim serialNumberString As String = New String(serialNumber, 0,
serialNumber.Length)
        If dnglError <> SmartDongle.SmartDnglError.UskOk Then
```

```
            System.Console.WriteLine(SmartDongle.GetErrorString(dnglError))
            ‘ handle error
        Else
            System.Console.WriteLine(“Serial Number: {0:G}”, serialNumberString)
        End If

        System.Console.WriteLine(“”)
        System.Console.WriteLine(“Press <Enter> to Exit”)
        System.Console.ReadLine()

    End Sub

End Module
```

d. SmartDongle Visual Basic 6 Support

Examples
The following examples require usk_vb.dll, included in the zipped image below. Edit the example code to use your unique key values, for example:

P1 = “0xec6cc589aefd1e75”
P2 = “0xfcec0a6a82747b3f”

1. Simple method to check for valid SmartDongle

```
VERSION 5.00
Begin VB.Form Form1
   Caption         =   “Form1”
   ClientHeight    =   1200
   ClientLeft      =   60
   ClientTop       =   345
   ClientWidth     =   7095
   LinkTopic       =   “Form1”
   ScaleHeight     =   1200
   ScaleWidth      =   7095
   StartUpPosition =   3  ‘Windows Default
   Begin VB.TextBox Text1
      BeginProperty Font
         Name           =   “MS Sans Serif”
         Size           =   9.75
         Charset        =   0
         Weight         =   700
         Underline      =   0   ‘False
         Italic         =   0   ‘False
         Strikethrough  =   0   ‘False
      EndProperty
      Height         =   495
      Left           =   240
      TabIndex       =   0
```

```
    Top           =   360
    Width         =   6495
  End
End
Attribute VB_Name = “Form1”
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
‘ Example 1: Simple program that verifies if a valid SmartDongle is plugged in.
‘
‘ MicroWorks, Inc.
‘ 2808 North Cole Road
‘ Boise, ID 83704
‘
‘ http://www.smartdongle.com

Option Explicit

Private Declare Function VBRead Lib “..\usk_vb” ( _
  ByVal P1 As String, _
  ByVal P2 As String, _
  ByVal address As Long, _
  ByVal Buffer As String, _
  ByVal size As Long _
) As Integer

Private Sub Form_Load()

Dim P1 As String
Dim P2 As String
Dim error As String
Dim status As Integer
Dim address As Integer
Dim size As Integer


Form1.Caption = “SmartDongle Example 1”
Form1.Show

‘
‘These are the SmartDongle Demo Keys
‘
P1 = “0xec6cc589aefd1e75”
P2 = “0xfcec0a6a82747b3f”

‘ Attempt SmartDongle access using demo keys
address = 0
size = 0
status = VBRead(P1, P2, address, 0, size)
```

```
error = Format(status)

If status Then
   Text1.Text = “FAILED: SmartDongle Access Failed with Error Code “ & error
Else
   Text1.Text = “SUCCESS: A valid SmartDongle is present”
End If

Form1.Refresh

End Sub
```

2. Write a secret to SmartDongle user memory

```
VERSION 5.00
Begin VB.Form Form1
  Caption        =   “Form1”
  ClientHeight   =   1260
  ClientLeft     =   60
  ClientTop      =   345
  ClientWidth    =   6900
  LinkTopic      =   “Form1”
  ScaleHeight    =   1260
  ScaleWidth     =   6900
  StartUpPosition =   3  ‘Windows Default
  Begin VB.TextBox Text1
    BeginProperty Font
      Name           =   “MS Sans Serif”
      Size           =   9.75
      Charset        =   0
      Weight         =   700
      Underline      =   0   ‘False
      Italic         =   0   ‘False
      Strikethrough  =   0   ‘False
    EndProperty
    Height         =   495
    Left           =   360
    TabIndex       =   0
    Top            =   360
    Width          =   6135
  End
End
Attribute VB_Name = “Form1”
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
‘
‘ Example2a: Write a secret to a SmartDongle
‘
```

```
‘ MicroWorks, Inc.
‘ 2808 North Cole Road
‘ Boise, ID 83704
‘
‘ http://www.smartdongle.com

Option Explicit

‘ For SmartDongle access
Private Const SecretAddress = 0
Private Const SecretSize = 16

‘  uskvb.dll functions
Private Declare Function LedRed Lib “..\usk_vb” () As Integer
Private Declare Function LedGreen Lib “..\usk_vb” () As Integer
Private Declare Function VBRead Lib “..\usk_vb” ( _
  ByVal P1 As String, _
  ByVal P2 As String, _
  ByVal address As Long, _
  ByVal buffer As String, _
  ByVal size As Long _
) As Integer
Private Declare Function VBWrite Lib “..\usk_vb” ( _
   ByVal P1 As String, _
   ByVal P2 As String, _
   ByVal address As Long, _
   ByVal buffer As String, _
   ByVal size As Long _
   ) As Integer


Private Sub Form_Load()

   Dim status As Integer   ‘ Function return value
   Dim errStr As String    ‘ Formatted error string
   Dim yourSecret As String

   ‘ SmartDongle Access vars
   Dim P1 As String        ‘ Key 1
   Dim P2 As String        ‘ Key 2
   Dim address As Integer  ‘ SmartDongle memory address
   Dim buffer As String    ‘ Input buffer for encrypted secret from SmartDongle
   Dim size As Integer     ‘ Input buffer size

   ‘ SmartDongle Demo Keys
   P1 = “0xec6cc589aefd1e75”
   P2 = “0xfcec0a6a82747b3f”

   ‘ Some SmartDongle secret
   yourSecret = “~JIKp=bm:-9+wn9}”
```

```
  buffer = String(SecretSize, vbNullChar)

  Form1.Caption = “Write Your Secret to SmartDongle”
  Form1.Show
  Form1.Refresh

  ‘ Indicate “Writing in progress”
  LedRed

  ‘ Attempt to write secret to SmartDongle
  address = SecretAddress
  size = SecretSize
  status = VBWrite(P1, P2, address, yourSecret, size)
  If status Then
    errStr = Format(status)
    Text1.Text = “FAILED: SmartDongle Access Failed with Error Code “ & errStr
    GoTo FAIL
  End If

  ‘ Attempt to read secret from SmartDongle
  status = VBRead(P1, P2, address, buffer, size)
  If status Then
    errStr = Format(status)
    Text1.Text = “FAILED: SmartDongle Access Failed with Error Code “ & errStr
    GoTo FAIL
  End If

  ‘Compare to expected secret
  If StrComp(Left$(buffer, SecretSize), Left$(yourSecret, SecretSize)) Then
    Text1.Text = “FAILED: Secret in SmartDongle Memory Does Not Match”
    GoTo FAIL
  End If

  ‘ Indicate “Secret Verified”
  LedGreen
  Text1.Text = “SUCCESS: Your Secret Has Been Written to SmartDongle”

FAIL:
  Form1.Refresh

End Sub
```

3. Read the secret written in Example2a.frm above

```
VERSION 5.00
Begin VB.Form Form1
  Caption        =   “Form1”
  ClientHeight   =   1260
  ClientLeft     =   60
  ClientTop      =   345
```

```
   ClientWidth     =   6900
   LinkTopic       =   “Form1”
   ScaleHeight     =   1260
   ScaleWidth      =   6900
   StartUpPosition =   3  ‘Windows Default
   Begin VB.TextBox Text1
     BeginProperty Font
       Name            =   “MS Sans Serif”
       Size            =   9.75
       Charset         =   0
       Weight          =   700
       Underline       =   0   ‘False
       Italic          =   0   ‘False
       Strikethrough   =   0   ‘False
     EndProperty
     Height          =   495
     Left            =   360
     TabIndex        =   0
     Top             =   360
     Width           =   6135
   End
End
Attribute VB_Name = “Form1”
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
‘ Example 2b: Read then verify a user secret from a SmartDongle.
‘
‘ MicroWorks, Inc.
‘ 2808 North Cole Road
‘ Boise, ID 83704
‘
‘ http://www.smartdongle.com

Option Explicit

‘ For SmartDongle access
Private Const SecretAddress = 0
Private Const SecretSize = 16

‘  uskvb.dll functions
Private Declare Function LedRed Lib “..\usk_vb” () As Integer
Private Declare Function LedGreen Lib “..\usk_vb” () As Integer
Private Declare Function VBRead Lib “..\usk_vb” ( _
  ByVal P1 As String, _
  ByVal P2 As String, _
  ByVal address As Long, _
  ByVal buffer As String, _
  ByVal size As Long _
```

```
) As Integer


Private Sub Form_Load()

  Dim status As Integer   ‘ Function return value
  Dim errStr As String    ‘ Formatted error string
  Dim yourSecret As String

  ‘ SmartDongle Access vars
  Dim P1 As String        ‘ Key 1
  Dim P2 As String        ‘ Key 2
  Dim address As Integer  ‘ SmartDongle memory address
  Dim buffer As String    ‘ Input buffer for encrypted secret from SmartDongle
  Dim size As Integer     ‘ Input buffer size

  ‘ SmartDongle Demo Keys
  P1 = “0xec6cc589aefd1e75”
  P2 = “0xfcec0a6a82747b3f”

  ‘ Expected SmartDongle secret
  yourSecret = “~JIKp=bm:-9+wn9}”
  buffer = String(SecretSize, vbNullChar)

  Form1.Caption = “Read and Decrypt Your SmartDongle Secret”
  Form1.Show
  Form1.Refresh

  ‘ Indicate “Verification in progress”
  LedRed

  ‘ Attempt to read encrypted secret from SmartDongle using demo keys
  address = SecretAddress
  size = SecretSize
  status = VBRead(P1, P2, address, buffer, size)
  If status Then
    errStr = Format(status)
    Text1.Text = “FAILED: SmartDongle Access Failed with Error Code “ & errStr
    GoTo FAIL
  End If

  ‘ Compare with your secret
  If StrComp(Left$(buffer, SecretSize), Left$(yourSecret, SecretSize)) Then
    Text1.Text = “FAILED: Invalid SmartDongle Secret”
    GoTo FAIL
  End If

  ‘ Indicate “Secret Verified”
  LedGreen
```

```
   Text1.Text = “SUCCESS: Valid SmartDongle Secret”

FAIL:
   Form1.Refresh

End Sub
```

4. Read SmartDongle serial number

```
VERSION 5.00
Begin VB.Form Form1
   Caption         =   “Form1”
   ClientHeight    =   1200
   ClientLeft      =   60
   ClientTop       =   345
   ClientWidth     =   7095
   LinkTopic       =   “Form1”
   ScaleHeight     =   1200
   ScaleWidth      =   7095
   StartUpPosition =   3  ‘Windows Default
   Begin VB.TextBox Text1
      BeginProperty Font
         Name            =   “MS Sans Serif”
         Size            =   9.75
         Charset         =   0
         Weight          =   700
         Underline       =   0   ‘False
         Italic          =   0   ‘False
         Strikethrough   =   0   ‘False
      EndProperty
      Height          =   495
      Left            =   240
      TabIndex        =   0
      Top             =   360
      Width           =   6495
   End
End
Attribute VB_Name = “Form1”
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
‘ Example 3: Read a SmartDongle serial number.
‘
‘ MicroWorks, Inc.
‘ 2808 North Cole Road
‘ Boise, ID 83704
‘
‘ http://www.smartdongle.com
```

```
Option Explicit

Private Const serialNumberLength = 12

Private Declare Function GetSerialNumber Lib "..\usk_vb" ( _
  ByVal serialNumber As String _
) As Integer

Private Sub Form_Load()

Dim serialNumber As String
Dim status As Integer

Form1.Caption = "SmartDongle Example 3"
Form1.Show

' Attempt to read SmartDongle serial number
serialNumber = String(serialNumberLength, vbNullChar)
status = GetSerialNumber(serialNumber)

If status Then
    Text1.Text = "FAILED: SmartDongle Access Failed with Error Code " & Error
Else
    Text1.Text = "SmartDongle serial number: " & serialNumber
End If

Form1.Refresh

End Sub
```

d. SmartDongle Visual Basic 6 Support

Examples
The following examples require usk_vb.dll, included in the zipped image below. Edit the example code to use your unique key values, for example:

```
P1 = "0xec6cc589aefd1e75"
P2 = "0xfcec0a6a82747b3f"
```

1. Simple method to check for valid SmartDongle

```
VERSION 5.00
Begin VB.Form Form1
   Caption         =   "Form1"
   ClientHeight    =   1200
   ClientLeft      =   60
   ClientTop       =   345
   ClientWidth     =   7095
   LinkTopic       =   "Form1"
   ScaleHeight     =   1200
```

```
   ScaleWidth      =   7095
   StartUpPosition =   3  'Windows Default
   Begin VB.TextBox Text1
      BeginProperty Font
         Name           =   "MS Sans Serif"
         Size           =   9.75
         Charset        =   0
         Weight         =   700
         Underline      =   0   'False
         Italic         =   0   'False
         Strikethrough  =   0   'False
      EndProperty
      Height         =   495
      Left           =   240
      TabIndex       =   0
      Top            =   360
      Width          =   6495
   End
End
Attribute VB_Name = "Form1"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
' Example 1: Simple program that verifies if a valid SmartDongle is plugged in.
'
' MicroWorks, Inc.
' 2808 North Cole Road
' Boise, ID 83704
'
' http://www.smartdongle.com

Option Explicit

Private Declare Function VBRead Lib "..\usk_vb" ( _
  ByVal P1 As String, _
  ByVal P2 As String, _
  ByVal address As Long, _
  ByVal Buffer As String, _
  ByVal size As Long _
) As Integer

Private Sub Form_Load()

Dim P1 As String
Dim P2 As String
Dim error As String
Dim status As Integer
Dim address As Integer
Dim size As Integer
```

```
Form1.Caption = “SmartDongle Example 1”
Form1.Show

‘
‘These are the SmartDongle Demo Keys
‘
P1 = “0xec6cc589aefd1e75”
P2 = “0xfcec0a6a82747b3f”

‘ Attempt SmartDongle access using demo keys
address = 0
size = 0
status = VBRead(P1, P2, address, 0, size)
error = Format(status)

If status Then
   Text1.Text = “FAILED: SmartDongle Access Failed with Error Code “ & error
Else
   Text1.Text = “SUCCESS: A valid SmartDongle is present”
End If

Form1.Refresh

End Sub
```

2. Write a secret to SmartDongle user memory

```
VERSION 5.00
BEGIN VB.FORM FORM1
  CAPTION        =   “FORM1”
  CLIENTHEIGHT    =   1260
  CLIENTLEFT      =   60
  CLIENTTOP       =   345
  CLIENTWIDTH     =   6900
  LINKTOPIC       =   “FORM1”
  SCALEHEIGHT     =   1260
  SCALEWIDTH      =   6900
  STARTUPPOSITION =   3  ‘WINDOWS DEFAULT
  BEGIN VB.TEXTBOX TEXT1
    BEGINPROPERTY FONT
      NAME           =   “MS SANS SERIF”
      SIZE           =   9.75
      CHARSET        =   0
      WEIGHT         =   700
      UNDERLINE      =   0   ‘FALSE
      ITALIC         =   0   ‘FALSE
      STRIKETHROUGH   =   0   ‘FALSE
    ENDPROPERTY
```

```
    HEIGHT        =   495
    LEFT          =   360
    TABINDEX      =   0
    TOP           =   360
    WIDTH         =   6135
  END
END
ATTRIBUTE VB_NAME = “FORM1”
ATTRIBUTE VB_GLOBALNAMESPACE = FALSE
ATTRIBUTE VB_CREATABLE = FALSE
ATTRIBUTE VB_PREDECLAREDID = TRUE
ATTRIBUTE VB_EXPOSED = FALSE
‘
‘ EXAMPLE2A: WRITE A SECRET TO A SMARTDONGLE
‘
‘ MICROWORKS, INC.
‘ 2808 NORTH COLE ROAD
‘ BOISE, ID 83704
‘
‘ HTTP://WWW.SMARTDONGLE.COM

OPTION EXPLICIT

‘ FOR SMARTDONGLE ACCESS
PRIVATE CONST SECRETADDRESS = 0
PRIVATE CONST SECRETSIZE = 16

‘  USKVB.DLL FUNCTIONS
PRIVATE DECLARE FUNCTION LEDRED LIB “..\USK_VB” () AS INTEGER
PRIVATE DECLARE FUNCTION LEDGREEN LIB “..\USK_VB” () AS INTEGER
PRIVATE DECLARE FUNCTION VBREAD LIB “..\USK_VB” ( _
  BYVAL P1 AS STRING, _
  BYVAL P2 AS STRING, _
  BYVAL ADDRESS AS LONG, _
  BYVAL BUFFER AS STRING, _
  BYVAL SIZE AS LONG _
) AS INTEGER
PRIVATE DECLARE FUNCTION VBWRITE LIB “..\USK_VB” ( _
   BYVAL P1 AS STRING, _
   BYVAL P2 AS STRING, _
   BYVAL ADDRESS AS LONG, _
   BYVAL BUFFER AS STRING, _
   BYVAL SIZE AS LONG _
   ) AS INTEGER


PRIVATE SUB FORM_LOAD()

   DIM STATUS AS INTEGER   ‘ FUNCTION RETURN VALUE
   DIM ERRSTR AS STRING    ‘ FORMATTED ERROR STRING
```

```
  DIM YOURSECRET AS STRING

  ‘ SMARTDONGLE ACCESS VARS
  DIM P1 AS STRING        ‘ KEY 1
  DIM P2 AS STRING        ‘ KEY 2
  DIM ADDRESS AS INTEGER  ‘ SMARTDONGLE MEMORY ADDRESS
  DIM BUFFER AS STRING    ‘ INPUT BUFFER FOR ENCRYPTED
SECRET FROM SMARTDONGLE
  DIM SIZE AS INTEGER     ‘ INPUT BUFFER SIZE

  ‘ SMARTDONGLE DEMO KEYS
  P1 = “0XEC6CC589AEFD1E75”
  P2 = “0XFCEC0A6A82747B3F”

  ‘ SOME SMARTDONGLE SECRET
  YOURSECRET = “~JIKP=BM:-9+WN9}”
  BUFFER = STRING(SECRETSIZE, VBNULLCHAR)

  FORM1.CAPTION = “WRITE YOUR SECRET TO SMARTDONGLE”
  FORM1.SHOW
  FORM1.REFRESH

  ‘ INDICATE “WRITING IN PROGRESS”
  LEDRED

  ‘ ATTEMPT TO WRITE SECRET TO SMARTDONGLE
  ADDRESS = SECRETADDRESS
  SIZE = SECRETSIZE
  STATUS = VBWRITE(P1, P2, ADDRESS, YOURSECRET, SIZE)
  IF STATUS THEN
    ERRSTR = FORMAT(STATUS)
    TEXT1.TEXT = “FAILED: SMARTDONGLE ACCESS FAILED WITH
ERROR CODE “ & ERRSTR
    GOTO FAIL
  END IF

  ‘ ATTEMPT TO READ SECRET FROM SMARTDONGLE
  STATUS = VBREAD(P1, P2, ADDRESS, BUFFER, SIZE)
  IF STATUS THEN
    ERRSTR = FORMAT(STATUS)
    TEXT1.TEXT = “FAILED: SMARTDONGLE ACCESS FAILED WITH
ERROR CODE “ & ERRSTR
    GOTO FAIL
  END IF

  ‘COMPARE TO EXPECTED SECRET
  IF STRCOMP(LEFT$(BUFFER, SECRETSIZE), LEFT$(YOURSECRET,
SECRETSIZE)) THEN
    TEXT1.TEXT = “FAILED: SECRET IN SMARTDONGLE MEMORY DOES
 NOT MATCH”
```

```
    GOTO FAIL
  END IF

  ‘ INDICATE “SECRET VERIFIED”
  LEDGREEN
  TEXT1.TEXT = “SUCCESS: YOUR SECRET HAS BEEN WRITTEN TO
 SMARTDONGLE”

FAIL:
  FORM1.REFRESH

END SUB
```

3. Read the secret written in Example2a.frm above.

```
VERSION 5.00
BEGIN VB.FORM FORM1
  CAPTION        =   “FORM1”
  CLIENTHEIGHT    =   1260
  CLIENTLEFT      =   60
  CLIENTTOP       =   345
  CLIENTWIDTH     =   6900
  LINKTOPIC       =   “FORM1”
  SCALEHEIGHT     =   1260
  SCALEWIDTH      =   6900
  STARTUPPOSITION =   3  ‘WINDOWS DEFAULT
  BEGIN VB.TEXTBOX TEXT1
    BEGINPROPERTY FONT
      NAME          =   “MS SANS SERIF”
      SIZE          =   9.75
      CHARSET        =   0
      WEIGHT         =   700
      UNDERLINE      =   0   ‘FALSE
      ITALIC         =   0   ‘FALSE
      STRIKETHROUGH   =   0   ‘FALSE
    ENDPROPERTY
    HEIGHT         =   495
    LEFT           =   360
    TABINDEX       =   0
    TOP            =   360
    WIDTH          =   6135
  END
END
ATTRIBUTE VB_NAME = “FORM1”
ATTRIBUTE VB_GLOBALNAMESPACE = FALSE
ATTRIBUTE VB_CREATABLE = FALSE
ATTRIBUTE VB_PREDECLAREDID = TRUE
ATTRIBUTE VB_EXPOSED = FALSE
‘ EXAMPLE 2B: READ THEN VERIFY A USER SECRET FROM A SMARTDONGLE.
‘
```

```
‘ MICROWORKS, INC.
‘ 2808 NORTH COLE ROAD
‘ BOISE, ID 83704
‘
‘ HTTP://WWW.SMARTDONGLE.COM

OPTION EXPLICIT

‘ FOR SMARTDONGLE ACCESS
PRIVATE CONST SECRETADDRESS = 0
PRIVATE CONST SECRETSIZE = 16

‘  USKVB.DLL FUNCTIONS
PRIVATE DECLARE FUNCTION LEDRED LIB “..\USK_VB” () AS INTEGER
PRIVATE DECLARE FUNCTION LEDGREEN LIB “..\USK_VB” () AS INTEGER
PRIVATE DECLARE FUNCTION VBREAD LIB “..\USK_VB” ( _
  BYVAL P1 AS STRING, _
  BYVAL P2 AS STRING, _
  BYVAL ADDRESS AS LONG, _
  BYVAL BUFFER AS STRING, _
  BYVAL SIZE AS LONG _
) AS INTEGER



PRIVATE SUB FORM_LOAD()

   DIM STATUS AS INTEGER   ‘ FUNCTION RETURN VALUE
   DIM ERRSTR AS STRING    ‘ FORMATTED ERROR STRING
   DIM YOURSECRET AS STRING

   ‘ SMARTDONGLE ACCESS VARS
   DIM P1 AS STRING        ‘ KEY 1
   DIM P2 AS STRING        ‘ KEY 2
   DIM ADDRESS AS INTEGER  ‘ SMARTDONGLE MEMORY ADDRESS
   DIM BUFFER AS STRING    ‘ INPUT BUFFER FOR ENCRYPTED SECRET
 FROM SMARTDONGLE
   DIM SIZE AS INTEGER     ‘ INPUT BUFFER SIZE

   ‘ SMARTDONGLE DEMO KEYS
   P1 = “0XEC6CC589AEFD1E75”
   P2 = “0XFCEC0A6A82747B3F”

   ‘ EXPECTED SMARTDONGLE SECRET
   YOURSECRET = “~JIKP=BM:-9+WN9}”
   BUFFER = STRING(SECRETSIZE, VBNULLCHAR)

   FORM1.CAPTION = “READ AND DECRYPT YOUR SMARTDONGLE SECRET”
   FORM1.SHOW
   FORM1.REFRESH
```

```
    ‘ INDICATE “VERIFICATION IN PROGRESS”
    LEDRED

    ‘ ATTEMPT TO READ ENCRYPTED SECRET FROM SMARTDONGLE USING
   DEMO KEYS
    ADDRESS = SECRETADDRESS
    SIZE = SECRETSIZE
    STATUS = VBREAD(P1, P2, ADDRESS, BUFFER, SIZE)
    IF STATUS THEN
      ERRSTR = FORMAT(STATUS)
      TEXT1.TEXT = “FAILED: SMARTDONGLE ACCESS FAILED WITH ERROR
  CODE “ & ERRSTR
      GOTO FAIL
    END IF

    ‘ COMPARE WITH YOUR SECRET
    IF STRCOMP(LEFT$(BUFFER, SECRETSIZE), LEFT$(YOURSECRET, SECRET-
SIZE)) THEN
      TEXT1.TEXT = “FAILED: INVALID SMARTDONGLE SECRET”
      GOTO FAIL
    END IF

    ‘ INDICATE “SECRET VERIFIED”
    LEDGREEN
    TEXT1.TEXT = “SUCCESS: VALID SMARTDONGLE SECRET”

  FAIL:
    FORM1.REFRESH

  END SUB
```

4. Read SmartDongle serial number

```
  VERSION 5.00
  Begin VB.Form Form1
    Caption         =   “Form1”
    ClientHeight    =   1200
    ClientLeft      =   60
    ClientTop       =   345
    ClientWidth     =   7095
    LinkTopic       =   “Form1”
    ScaleHeight     =   1200
    ScaleWidth      =   7095
    StartUpPosition =   3  ‘Windows Default
    Begin VB.TextBox Text1
      BeginProperty Font
        Name            =   “MS Sans Serif”
        Size            =   9.75
        Charset         =   0
```

```
      Weight         =   700
      Underline      =   0   'False
      Italic         =   0   'False
      Strikethrough  =   0   'False
    EndProperty
    Height         =   495
    Left           =   240
    TabIndex       =   0
    Top            =   360
    Width          =   6495
  End
End
Attribute VB_Name = "Form1"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
' Example 3: Read a SmartDongle serial number.
'
' MicroWorks, Inc.
' 2808 North Cole Road
' Boise, ID 83704
'
' http://www.smartdongle.com

Option Explicit

Private Const serialNumberLength = 12

Private Declare Function GetSerialNumber Lib "..\usk_vb" ( _
  ByVal serialNumber As String _
) As Integer

Private Sub Form_Load()

Dim serialNumber As String
Dim status As Integer

Form1.Caption = "SmartDongle Example 3"
Form1.Show

' Attempt to read SmartDongle serial number
serialNumber = String(serialNumberLength, vbNullChar)
status = GetSerialNumber(serialNumber)

If status Then
    Text1.Text = "FAILED: SmartDongle Access Failed with Error Code " & Error
Else
    Text1.Text = "SmartDongle serial number: " & serialNumber
End If
```

```
Form1.Refresh

End Sub
```

II. Linux, MacOS X (Power PC and Intel), FreeBSD, OpenBSD

A. SmartDongle Userspace Interface using Libusb

This code has been tested on the following systems:
Linux (2.4.x and 2.6.x)
Mac OS X (both Intel Mac and PowerPC Mac)

current code untested on Solaris and FreeBSD

1. Example 1: Simple method to check for valid SmartDongle

```
/*
  example1.c:  Check if a valid SmartDongle is plugged in.

  MicroWorks, Inc.
  2808 North Cole Road
  Boise, ID 83704

  www.smartdongle.com


  ***** BEGIN LICENSE BLOCK *****

  Version: MPL 1.1/GPL 2.0/LGPL 2.1

  The contents of this file are subject to the Mozilla Public License
  Version 1.1 (the “License”); you may not use this file except in
  compliance with the License. You may obtain a copy of the License at
  http://www.mozilla.org/MPL/

  Software distributed under the License is distributed on an “AS IS” basis,
  WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License
  for the specific language governing rights and limitations under the
  License.

  The Original Code is MicroWorks, Inc. code.

  The Initial Developer of the Original Code is MicroWorks, Inc. Portions
  created by MicroWorks, Inc. are Copyright (C) 2006
  MicroWorks, Inc. All Rights Reserved.

  Contributor(s):
```

```
   Alternatively, the contents of this file may be used under the terms of
   either the GNU General Public License Version 2 or later (the “GPL”), or
   the GNU Lesser General Public License Version 2.1 or later (the “LGPL”),
   in which case the provisions of the GPL or the LGPL are applicable instead
   of those above. If you wish to allow use of your version of this file only
   under the terms of either the GPL or the LGPL, and not to allow others to
   use your version of this file under the terms of the MPL, indicate your
   decision by deleting the provisions above and replace them with the notice
   and other provisions required by the GPL or the LGPL. If you do not delete
   the provisions above, a recipient may use your version of this file under
   the terms of any one of the MPL, the GPL or the LGPL.

   ***** END LICENSE BLOCK *****
*/
#include <stdio.h>
#include <string.h>
#include “smartdongle.h”

int main ( int argc, char **argv) {

    int error;
    char errorString[SMARTDNGL_ERROR_STRING_LENGTH_MAX];

    /* SmartDongle demo keys. */
    unsigned long long P1 = 0xec6cc589aefd1e75ULL;
    unsigned long long P2 = 0xfcec0a6a82747b3fULL;

    /*
       Simply check if a valid SmartDongle is present.
    */
    error =  SmartDongleRead(&P1, &P2, 0, NULL, 0);
    if (error) {

       /*
          If error code is 41 or 42, the problem may be that the mode of the
          dynamic SmartDongle device files in /proc/bus/usb/ and
          /dev/bus/usb/ do not allow user access.

          As of linux kernel version 2.6.x, permissions can be set with hal
          facilities.  Refer to HAL/README.txt
       */

       SmartDongleGetErrorString(errorString, error);
       printf(“FAIL: error %d, %s\n”, error, errorString);

       return 1;
    }

    printf(“SUCCESS: Valid SmartDongle present\n”);
```

```
    return 0;
}
```

2. Example 2a: Write some application data to a SmartDongle

```
/*
    example2.c:  Write some application data to a SmartDongle.

    MicroWorks, Inc.
    2808 North Cole Road
    Boise, ID 83704

    www.smartdongle.com


    ***** BEGIN LICENSE BLOCK *****

    Version: MPL 1.1/GPL 2.0/LGPL 2.1

    The contents of this file are subject to the Mozilla Public License
    Version 1.1 (the "License"); you may not use this file except in
    compliance with the License. You may obtain a copy of the License at
    http://www.mozilla.org/MPL/

    Software distributed under the License is distributed on an "AS IS" basis,
    WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License
    for the specific language governing rights and limitations under the
    License.

    The Original Code is MicroWorks, Inc. code.

    The Initial Developer of the Original Code is MicroWorks, Inc. Portions
    created by MicroWorks, Inc. are Copyright (C) 2008
    MicroWorks, Inc. All Rights Reserved.

    Contributor(s):

    Alternatively, the contents of this file may be used under the terms of
    either the GNU General Public License Version 2 or later (the "GPL"), or
    the GNU Lesser General Public License Version 2.1 or later (the "LGPL"),
    in which case the provisions of the GPL or the LGPL are applicable instead
    of those above. If you wish to allow use of your version of this file only
    under the terms of either the GPL or the LGPL, and not to allow others to
    use your version of this file under the terms of the MPL, indicate your
    decision by deleting the provisions above and replace them with the notice
    and other provisions required by the GPL or the LGPL. If you do not delete
    the provisions above, a recipient may use your version of this file under
    the terms of any one of the MPL, the GPL or the LGPL.

    ***** END LICENSE BLOCK *****
```

```
*/
#include <stdio.h>
#include <string.h>
#include “smartdongle.h”


/*
   Your application data here.
*/
#define YOUR_DATA “This is data written to the SmartDongle, it can be
serial numbers, program parameters, etc.  Note that this data can be read by
 observing activity on the USB bus. Refer to example3a.c and example3b.c for
example of encrypting your application data.”

int main ( int argc, char **argv) {


   /* SmartDongle demo keys. */
   unsigned long long P1 = 0xec6cc589aefd1e75ULL;
   unsigned long long P2 = 0xfcec0a6a82747b3fULL;

   unsigned short address;
   unsigned char data[] = YOUR_DATA;
   unsigned char *in;
   char errorString[SMARTDNGL_ERROR_STRING_LENGTH_MAX];
   int error = SMARTDNGL_ERR_OK;
   int size;


   /*
      Turn SmartDongle Red LED on to indicate write in progress.
   */
   SmartDongleLedRed();

   /*
      Write application data to SmartDongle

      If error code is 41 or 42, the problem may be that the mode of the
      dynamic SmartDongle device files in /proc/bus/usb/ and /dev/bus/usb/
      do not allow user access.

      As of linux kernel version 2.6.x, permissions can be set with hal
      facilities.  Refer to HAL/README.txt
   */
   address = 0;
   size = strlen((char *)data) + 1; // add 1 for terminating null charater
   error = SmartDongleWrite(&P1, &P2, address, data, size);
   if (error) {
      SmartDongleGetErrorString(errorString, error);
      printf(“Write Failed %d, %s\n”, error, errorString);
```

```
        return 1;
    }

    /*
        Read the data back

        If error code is 41 or 42, the problem may be that the mode of the
        dynamic SmartDongle device files /proc/bus/usb/ and /dev/bus/usb/
        does not allow user access.

        As of linux kernel version 2.6.x, permissions can be set with hal
        facilities.  Refer to HAL/README.txt
    */
    in = (unsigned char *)malloc(size);
    if (in == NULL) {
        error =  SMARTDNGL_ERR_MEM;
        SmartDongleGetErrorString(errorString, error);
        printf("Error %d, %s\n", error, errorString);
        return 1;
    }

    error =  SmartDongleRead(&P1, &P2, address, in, size);
    if (error) {
        SmartDongleGetErrorString(errorString, error);
        printf("Read Failed %d, %s\n", error, errorString);

        free(in);
        return 1;
    }

    // Compare data to expected user data
    if ( memcmp(in, data, size) != 0) {
        printf ("Error writing data to SmartDongle, miscompare\n");
    } else {
        printf ("Successfully wrote your application data to SmartDongle\n");

        // Inicate success
        SmartDongleLedGreen();
    }

    free(in);
    return 0;
}
```

3. Example 2b: Read the application data written in Example2a

```
/*
    example2b.c: Read application data written to SmartDongle in example2a.c

    MicroWorks, Inc.
```

2808 North Cole Road
Boise, ID 83704

www.smartdongle.com

***** BEGIN LICENSE BLOCK *****

Version: MPL 1.1/GPL 2.0/LGPL 2.1

***** END LICENSE BLOCK *****
*/

```
#include <stdio.h>
#include <string.h>
#include “smartdongle.h”

/*
   Your application data here.
*/
#define YOUR_DATA “This is data written to the SmartDongle, it can be
```

serial numbers, program parameters, etc. Note that this data can be read by observing activity on the USB bus. Refer to example3a.c and example3b.c for example of encrypting your application data."

```
int main ( int argc, char **argv) {

    /* SmartDongle demo keys. */
    unsigned long long P1 = 0xec6cc589aefd1e75ULL;
    unsigned long long P2 = 0xfcec0a6a82747b3fULL;

    unsigned char data[] = YOUR_DATA;
    unsigned char *in;
    unsigned short address;
    int length;
    char errorString[SMARTDNGL_ERROR_STRING_LENGTH_MAX];
    int error = SMARTDNGL_ERR_OK;


    /*
        Allocate input buffer for expected application data size
        plus one for C string null terminator.
    */
    length = strlen((char *)data) + 1;
    in = (unsigned char *)malloc(length);
    if (in == NULL) {
        error = SMARTDNGL_ERR_MEM;
        SmartDongleGetErrorString(errorString, error);
        printf("%s\n", errorString);
        return 1;
    }

    /*
        Read the application data written in example2a.c
    */
    address = 0;
    error = SmartDongleRead(&P1, &P2, address, in, length);
    if (error) {
        SmartDongleGetErrorString(errorString, error);
        printf("SmartDongle Read Failed %d, %s\n", error, errorString);
        return 1;
    }

    /*
       This exercise simply compares the data read from the SmartDongle with
       the expected application data.  A developer could do any number of
       things with the data.
    */
    if (memcmp(in, data, length) == 0) {
        printf("Success: SmartDongle has the expected application data\n");
    } else {
```

```
        printf("SmartDongle does not have the expected application data\n");
        return 1;
    }

    return 0;
}
```

If you store data on a SmartDongle you may wish to encrypt the data transmitted over the USB bus.

Examples 3a and 3b below use AES encryption. A developer could use example3a.c as a utility to write passwords, serial numbers, etc to a SmartDongle that can be read back and used in an application as demonstrated by example3b.c

4. Example 3a: Encrypt a user string using AES and write this string to a SmartDongle

```
/*
    example3a.c: Write then verify an encrypted secret to a SmartDongle.

    MicroWorks, Inc.
    2808 North Cole Road
    Boise, ID 83704

    www.smartdongle.com


    ***** BEGIN LICENSE BLOCK *****

    Version: MPL 1.1/GPL 2.0/LGPL 2.1

    The contents of this file are subject to the Mozilla Public License
    Version 1.1 (the "License"); you may not use this file except in
    compliance with the License. You may obtain a copy of the License at
    http://www.mozilla.org/MPL/

    Software distributed under the License is distributed on an "AS IS" basis,
    WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License
    for the specific language governing rights and limitations under the
    License.

    The Original Code is MicroWorks, Inc. code.

    The Initial Developer of the Original Code is MicroWorks, Inc. Portions
    created by MicroWorks, Inc. are Copyright (C) 2008
    MicroWorks, Inc. All Rights Reserved.

    Contributor(s):
```

```
   Alternatively, the contents of this file may be used under the terms of
   either the GNU General Public License Version 2 or later (the “GPL”), or
   the GNU Lesser General Public License Version 2.1 or later (the “LGPL”),
   in which case the provisions of the GPL or the LGPL are applicable instead
   of those above. If you wish to allow use of your version of this file only
   under the terms of either the GPL or the LGPL, and not to allow others to
   use your version of this file under the terms of the MPL, indicate your
   decision by deleting the provisions above and replace them with the notice
   and other provisions required by the GPL or the LGPL. If you do not delete
   the provisions above, a recipient may use your version of this file under
   the terms of any one of the MPL, the GPL or the LGPL.

   ***** END LICENSE BLOCK *****
*/
#include <stdio.h>
#include <string.h>
#include “smartdongle.h”
#include “aes.h”

int main ( int argc, char **argv) {

   /* SmartDongle demo keys. */
   unsigned long long P1 = 0xec6cc589aefd1e75ULL;
   unsigned long long P2 = 0xfcec0a6a82747b3fULL;

   /*
      This is a demo 128 bit encryption key,
      chose some another 16 character key for your project.
   */
   unsigned char AESkey[] = “7obdgEn2’TH>]SE|”;

   /*
      Demo secret
   */
   unsigned char plainText[] = “This is the secret written to SmartDongle,
 it can be passwords, serial numbers, program parameters, etc”;

   char errorString[SMARTDNGL_ERROR_STRING_LENGTH_MAX];
   SmartDongleCipher *cipher;
   SmartDongleCipher cipherInput;
   unsigned char *input;
   int size, error;
   unsigned short address;



   /* Indicate write in progress */
   SmartDongleLedRed();
```

```
/*
   Encrypt your secret using AES with a block size of 128
*/
cipher =  SmartDongleEncode(
   plainText,
   strlen((char *)plainText) + 1,
   128,
   AESkey);
if (cipher == NULL) {

   error = SMARTDNGL_ERR_UNKNOWN;
   printf("Error encoding user secret: illegal AES block size?\n");
   return 1;
}


/*
   Write your encrypted data to SmartDongle.

   If error code is 41 or 42, the problem may be that the mode of the
   dynamic SmartDongle device files in /proc/bus/usb/ and /dev/bus/usb/
   do not allow user access.

   As of linux kernel version 2.6.x, permissions can be set with hal
   facilities.  Refer to HAL/README.txt
*/
address = 0;
size = (cipher->AESblockSize / 8) * cipher->numberOfBlocks;
error = SmartDongleWrite(&P1, &P2, address, cipher->data, size);
if (error) {

   SmartDongleGetErrorString(errorString, error);
   printf("Write Failed %d, %s\n", error, errorString);
   free(cipher->data);
   free(cipher);
   return 1;
}


/*
   Allocate input for cipher read from SmartDongle
*/
input = (unsigned char *)malloc(size);
if (input == NULL) {

   error =  SMARTDNGL_ERR_MEM;
   SmartDongleGetErrorString(errorString, error);
   printf("Error %d, %s\n", error, errorString);
   free(cipher->data);
   free(cipher);
```

```
    return 1;
}


/*
    Read encrypted data back from SmartDongle

    If error code is 41 or 42, the problem may be that the mode of the
    dynamic SmartDongle device files /proc/bus/usb/ and /dev/bus/usb/
    does not allow user access.

    As of linux kernel version 2.6.x, permissions can be set with hal
    facilities.  Refer to HAL/README.txt
*/
address = 0;
error = SmartDongleRead(&P1, &P2, address, input, size);
if (error) {

    SmartDongleGetErrorString(errorString, error);
    printf("Read Failed %d, %s\n", error, errorString);
    free(cipher->data);
    free(cipher);
    free(input);
    return 1;
}


/*
    Decrypt
*/
cipherInput.AESblockSize = cipher->AESblockSize;
cipherInput.numberOfBlocks = cipher->numberOfBlocks;
cipherInput.data = input;
SmartDongleDecode(&cipherInput, AESkey);


/*
    Compare decrypted cipher to expected user data.
    Compare using orignal secret length since the decryped cipher may have
    been padded with zeroes.
*/
if (memcmp(cipherInput.data, plainText, strlen((char *)plainText)) == 0) {

    printf ("Success: Your encrypted secret was written to SmartDongle\n");

    /* Indicate success */
    SmartDongleLedGreen();
} else {

    printf ("Fail: Data read from SmartDongle did not match\n");
```

```
    }


    free(cipher->data);
    free(cipher);
    free(input);
    return 0;
}
```

5. Example 3b: Read and decrypt the secret written in Example3a

```
/*
    example3b.c: Verify a secret on a SmartDongle

    MicroWorks, Inc.
    2808 North Cole Road
    Boise, ID 83704

    www.smartdongle.com


    ***** BEGIN LICENSE BLOCK *****

    Version: MPL 1.1/GPL 2.0/LGPL 2.1

    The contents of this file are subject to the Mozilla Public License
    Version 1.1 (the “License”); you may not use this file except in
    compliance with the License. You may obtain a copy of the License at
    http://www.mozilla.org/MPL/

    Software distributed under the License is distributed on an “AS IS” basis,
    WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License
    for the specific language governing rights and limitations under the
    License.

    The Original Code is MicroWorks, Inc. code.

    The Initial Developer of the Original Code is MicroWorks, Inc. Portions
    created by MicroWorks, Inc. are Copyright (C) 2008
    MicroWorks, Inc. All Rights Reserved.

    Contributor(s):

    Alternatively, the contents of this file may be used under the terms of
    either the GNU General Public License Version 2 or later (the “GPL”), or
    the GNU Lesser General Public License Version 2.1 or later (the “LGPL”),
    in which case the provisions of the GPL or the LGPL are applicable
    instead of those above. If you wish to allow use of your version of this file only
    under the terms of either the GPL or the LGPL, and not to allow others to
    use your version of this file under the terms of the MPL, indicate your
```

```
   decision by deleting the provisions above and replace them with the notice
   and other provisions required by the GPL or the LGPL. If you do not delete
   the provisions above, a recipient may use your version of this file under
   the terms of any one of the MPL, the GPL or the LGPL.

   ***** END LICENSE BLOCK *****
*/
#include <stdio.h>
#include <string.h>
#include “smartdongle.h”
#include “aes.h”

int main ( int argc, char **argv) {

   /* SmartDongle demo keys. */
   unsigned long long P1 = 0xec6cc589aefd1e75ULL;
   unsigned long long P2 = 0xfcec0a6a82747b3fULL;

   /*
      This is a demo 128 bit encryption key,
      chose some another 16 character key for your project.
   */
   unsigned char AESkey[] = “7obdgEn2’TH>]SE|”;

   /*
      Demo secret
   */
   unsigned char plainText[] = “This is the secret written to SmartDongle,
 it can be passwords, serial numbers, program parameters, etc”;

   char errorString[SMARTDNGL_ERROR_STRING_LENGTH_MAX];
   SmartDongleCipher cipher;
   unsigned char *input;
   int plainTextLength;
   int size, error;
   unsigned short address;


   /*
      Calculate the number of AES blocks to be read from SmartDongle,
      rounding block count up to nearest block size.
   */
   plainTextLength = strlen((char *)plainText) + 1; // length + null character
   cipher.AESblockSize = 128;
   cipher.numberOfBlocks =
      (plainTextLength * 8) / cipher.AESblockSize +
      ((plainTextLength * 8 % cipher.AESblockSize > 0) ? 1 : 0);


   /*
```

```
   Allocate data string for cipher read from SmartDongle
*/
size = (cipher.AESblockSize / 8) * cipher.numberOfBlocks;
input = (unsigned char *)malloc(size);
if (input  == NULL) {

   error =  SMARTDNGL_ERR_MEM;
   SmartDongleGetErrorString(errorString, error);
   printf("Error %d, %s\n", error, errorString);
   return 1;
}


/*
   Read encrypted data back from SmartDongle

   If error code is 41 or 42, the problem may be that the mode of the
   dynamic SmartDongle device files /proc/bus/usb/ and /dev/bus/usb/
   does not allow user access.

   As of linux kernel version 2.6.x, permissions can be set with hal
   facilities.  Refer to HAL/README.txt
*/
address = 0;
error =  SmartDongleRead(&P1, &P2, address, input, size);
if (error) {

   SmartDongleGetErrorString(errorString, error);
   printf("Read Failed %d, %s\n", error, errorString);
   free(input);
   return 1;
}


/*
   Decrypt
*/
cipher.data = input;
SmartDongleDecode(&cipher, AESkey);


/*
   Compare to original secret, using orignal secret length.

   The data comparison here is just an exercise, the developer can do
   any number of things with the decrypted user data here.
*/
if (memcmp(cipher.data, plainText, strlen((char *)plainText)) == 0) {

   printf ("Success: Your encrypted secret was found on SmartDongle\n");
```

```
    } else {

        printf ("Fail: Your secret was not found on SmartDongle\n");
    }

    free(input);
    return 0;
}
```

6. Read SmartDongle serial number

```
/*
   example4.c: Read SmartDongle serial number and memory size

   MicroWorks, Inc.
   2808 North Cole Road
   Boise, ID 83704

   www.smartdongle.com


   ***** BEGIN LICENSE BLOCK *****

   Version: MPL 1.1/GPL 2.0/LGPL 2.1

   The contents of this file are subject to the Mozilla Public License
   Version 1.1 (the "License"); you may not use this file except in
   compliance with the License. You may obtain a copy of the License at
   http://www.mozilla.org/MPL/

   Software distributed under the License is distributed on an "AS IS" basis,
   WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License
   for the specific language governing rights and limitations under the
   License.

   The Original Code is MicroWorks, Inc. code.

   The Initial Developer of the Original Code is MicroWorks, Inc. Portions
   created by MicroWorks, Inc. are Copyright (C) 2008
   MicroWorks, Inc. All Rights Reserved.

   Contributor(s):

   Alternatively, the contents of this file may be used under the terms of
   either the GNU General Public License Version 2 or later (the "GPL"), or
   the GNU Lesser General Public License Version 2.1 or later (the "LGPL"),
   in which case the provisions of the GPL or the LGPL are applicable instead
   of those above. If you wish to allow use of your version of this file only
   under the terms of either the GPL or the LGPL, and not to allow others to
   use your version of this file under the terms of the MPL, indicate your
```

```
   decision by deleting the provisions above and replace them with the notice
   and other provisions required by the GPL or the LGPL. If you do not delete
   the provisions above, a recipient may use your version of this file under
   the terms of any one of the MPL, the GPL or the LGPL.

   ***** END LICENSE BLOCK *****
*/
#include <stdio.h>
#include <string.h>
#include “smartdongle.h”

int main ( int argc, char **argv) {

    int error;
    int userMemorySize;
    char sn[SMARTDNGL_SN_LENGTH * 10 + 1];
    char errorString[SMARTDNGL_ERROR_STRING_LENGTH_MAX];

    /* SmartDongle demo keys. */
    unsigned long long P1 = 0xec6cc589aefd1e75ULL;
    unsigned long long P2 = 0xfcec0a6a82747b3fULL;


    /*
       Note:

       If error code 41 or 42 is returned from functions below, the problem
       may be that the mode of the dynamic SmartDongle device files in
       /proc/bus/usb/ and /dev/bus/usb/ do not allow user access.

       As of linux kernel version 2.6.x, permissions can be set with hal
       facilities.  Refer to HAL/README.txt
    */


    /*
       Retrieve SmartDongle serial number
    */
    error = SmartDongleGetSerialNumber(sn);
    if (error) {

       SmartDongleGetErrorString(errorString, error);
       printf(“FAIL: error %d, %s\n”, error, errorString);
    } else {

       printf(“Serial Number: %s\n”, sn);
    }

    /*
       Retrieve SmartDongle memory size that is available to user
```

```
    */
    userMemorySize = SmartDongleGetEepromSize(&P1, &P2);
    if (userMemorySize < 0) {

        SmartDongleGetErrorString(errorString, -userMemorySize);
        printf("FAIL: error %d, %s\n", -userMemorySize, errorString);
    } else {

        printf("User memory size: %d\n", userMemorySize);
    }

  return 0;
}
```

b. SmartDongle Java Native Interface for libusb

This code has been tested on the following systems:
Linux (2.4.x and 2.6.x)
Mac OS X (both Intel Mac and PowerPC Mac)

current code untested on Solaris and FreeBSD

Example 1: Simple method to check for valid SmartDongle

```
/**
 * Example1.java   Check if a valid SmartDongle is plugged in.

   MicroWorks, Inc.
   2808 North Cole Road
   Boise, ID 83704

   www.smartdongle.com


   ***** BEGIN LICENSE BLOCK *****

   Version: MPL 1.1/GPL 2.0/LGPL 2.1

   The contents of this file are subject to the Mozilla Public License
   Version 1.1 (the "License"); you may not use this file except in
   compliance with the License. You may obtain a copy of the License at
   http://www.mozilla.org/MPL/

   Software distributed under the License is distributed on an "AS IS" basis,
   WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License
   for the specific language governing rights and limitations under the
   License.

   The Original Code is MicroWorks, Inc. code.
```

*/

```
import java.security.MessageDigest;

public class Example1 {

  private static int error;

  // Your keys here
  private final static long p1 = 0xEC6CC589AEFD1E75L;
  private final static long p2 = 0xFCEC0A6A82747B3FL;

  // SmartDongle JNI library.
  private final static String apiJniName = “libuskjni.so”;

  // The relative path to uskjni.dll from your main class.
  private final static String apiJniRelativePath = “.”;


  public static void main(String args[]) {

    String fileSeparator = System.getProperty(“file.separator”);
    String apiJniAbsolutePath = System.getProperty(“user.dir”)
          + fileSeparator
          + apiJniRelativePath
          + fileSeparator
          + apiJniName;

    if (!smartDongleInitialize(apiJniAbsolutePath)) {
      // FAIL exiting...
```

```
      return;
    }

    error = smartDongleRead(p1, p2, 0, new byte[0], 0);
    if (error != 0) {
      // FAIL exiting...
      return;
     }




    /*

      Your main class code goes here

    */
    System.out.println("Success: Will execute main class");




  }



  /*
   *  SmartDongle methods here
   */
  private static native int smartDongleRead(
    long p1, long p2, int addr, byte[] data, int size);

  private static boolean smartDongleInitialize (String apiJniPath) {

// AWK_USKJNI_SIG_BEGIN
byte sig[] = {80, -60, -90, -20, -116, -36, -9, -72, -53, -96, 127, -105, -122,
-21, 47, 88, 50, -89, -80, 49, };
// AWK_USKJNI_SIG_END

    MessageDigest md = null;

    /*
     * Check uskjni SHA fingerprint
     */
    try {
      md = MessageDigest.getInstance("SHA");

      java.io.FileInputStream in = new java.io.FileInputStream(apiJniPath);
      int len;
```

```
      byte[] input = new byte[512];

      while ((len = in.read(input)) > 0) {
        md.update(input, 0, len);
      }

      byte[] digest = md.digest();

      // should be 20 bytes, 160 bits long
      if (digest.length != sig.length) {
        return false;
      }

      // Check uskjni.dll SHA fingerprint
      for (int i = 0; i<digest.length; i++) {

        if (digest[i] != sig[i]) {
          return false;
        }
      }

      System.load(apiJniPath);
    }
    catch (java.io.IOException ioe) {
      System.out.println("Caught Exception:  " + ioe);
    }
    catch (java.security.NoSuchAlgorithmException nsae) {
      System.out.println("Caught Exception:  " + nsae);
    }

    return true;
  }
}
```

2. Example 2a: Write some application data to a SmartDongle

```
/**
    Example2a.java  Write some application data to a SmartDongle.

    MicroWorks, Inc.
    2808 North Cole Road
    Boise, ID 83704

    www.smartdongle.com


    ***** BEGIN LICENSE BLOCK *****

    Version: MPL 1.1/GPL 2.0/LGPL 2.1
```

*/

```
import java.security.MessageDigest;
import java.lang.reflect.Array;


public class Example2a {

  // Your keys here
  private final static long p1 = 0xEC6CC589AEFD1E75L;
  private final static long p2 = 0xFCEC0A6A82747B3FL;

  private static int error;

  //The relative path to uskjni.dll from your main class.
  private final static String apiJniRelativePath = ".";

  // SmartDongle JNI library.
```

```
  private final static String apiJniName = “libuskjni.so”;

  // Some application data
  private final static String data = “This is data written to the SmartDongle, it can be
serial numbers, program parameters, etc.  Note that this data can be read by
observing activity on the USB bus. Refer to Example3a and Example3b for
example of encrypting your
 application data.”;


  public static void main(String args[]) {

    String fileSeparator = System.getProperty(“file.separator”);
    String apiJniAbsolutePath =
      System.getProperty(“user.dir”) +
      fileSeparator +
      apiJniRelativePath +
      fileSeparator +
      apiJniName;

    if (!smartDongleInitialize(apiJniAbsolutePath)) {
      System.out.println(“SmartDongle JNI initialisation failed”);
      return;
    }

    // Indicate write in progress
    smartDongleLedRed();

    // Write application data to a SmartDongle
    byte[] dataBytes = data.getBytes();
    int address = 0;
    int dataLength = Array.getLength(dataBytes);
    error = smartDongleWrite(p1,
                             p2,
                             address,
                             dataBytes,
                             dataLength);
    if (error != 0) {
      System.out.println(
        “Error writing to SmartDongle: “ + error + “ exiting...”);
      return;
    }

    // Read application data
    byte[] inBytes = new byte[dataLength];
    error = smartDongleRead(p1,
                            p2,
                            address,
                            inBytes,
                            dataLength);
```

```
  if (error != 0) {
    System.out.println(
      “Error reading from SmartDongle: “ + error + “ exiting...”);
    return;
  }

  // Compare data read from SmartDongle to check data for integrity.
  int i;
  for (i = 0; i < dataLength; i++) {
    if (inBytes[i] != dataBytes[i]) {
      break;
    }
  }

  if (i == dataLength) {

    // Indicate success
    smartDongleLedGreen();

    System.out.println(“Success writing application data of length “
                       + Array.getLength(dataBytes));
  }
  else
  {
    System.out.println(“Failure writing application data, mismatch”);
  }
}


/*
 *  SmartDongle native methods here
 */
private static native int smartDongleRead(long p1,
                                          long p2,
                                          int addr,
                                          byte[] data,
                                          int size);

private static native int smartDongleWrite(long p1,
                                           long p2,
                                           int addr,
                                           byte[] data,
                                           int size);

private static native int smartDongleLedRed();
private static native int smartDongleLedGreen();


private static boolean smartDongleInitialize(String apiJniPath) {
```

```
// AWK_USKJNI_SIG_BEGIN
byte sig[] = {-80, 63, -50, 96, -115, 91, 123, 99, -40, 118, -6, 71, 14, -5, 77,
 -20, 67, -6, -102, 3, };
// AWK_USKJNI_SIG_END

    MessageDigest md = null;

    /*
     * Check uskjni SHA fingerprint
     */
    try {
      md = MessageDigest.getInstance("SHA");

      java.io.FileInputStream in = new java.io.FileInputStream(apiJniPath);
      int len;

      byte[] input = new byte[512];

      while ((len = in.read(input)) > 0) {
        md.update(input, 0, len);
      }

      byte[] digest = md.digest();

      // should be 20 bytes, 160 bits long
      if (digest.length != sig.length) {
        return false;
      }

      // Check uskjni.dll SHA fingerprint
      for (int i = 0; i<digest.length; i++) {
        if (digest[i] != sig[i]) {
          return false;
        }
      }

      System.load(apiJniPath);
    }
    catch (java.io.IOException ioe) {
      System.out.println("Caught Exception:  " + ioe);
    }
    catch (java.security.NoSuchAlgorithmException nsae) {
      System.out.println("Caught Exception:  " + nsae);
    }

    return true;
  }
}
```

3. Example 2b: Read the application data written in Example2a

```
/**
   Example2b.java  Read application data written to SmartDongle in Example2a

   MicroWorks, Inc.
   2808 North Cole Road
   Boise, ID 83704

   www.smartdongle.com


   ***** BEGIN LICENSE BLOCK *****

   Version: MPL 1.1/GPL 2.0/LGPL 2.1

   The contents of this file are subject to the Mozilla Public License
   Version 1.1 (the “License”); you may not use this file except in
   compliance with the License. You may obtain a copy of the License at
   http://www.mozilla.org/MPL/

   Software distributed under the License is distributed on an “AS IS” basis,
   WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License
   for the specific language governing rights and limitations under the
   License.

   The Original Code is MicroWorks, Inc. code.

   The Initial Developer of the Original Code is MicroWorks, Inc. Portions
   created by MicroWorks, Inc. are Copyright (C) 2008
   MicroWorks, Inc. All Rights Reserved.

   Contributor(s):

   Alternatively, the contents of this file may be used under the terms of
   either the GNU General Public License Version 2 or later (the “GPL”), or
   the GNU Lesser General Public License Version 2.1 or later (the “LGPL”),
   in which case the provisions of the GPL or the LGPL are applicable instead
   of those above. If you wish to allow use of your version of this file only
   under the terms of either the GPL or the LGPL, and not to allow others to
   use your version of this file under the terms of the MPL, indicate your
   decision by deleting the provisions above and replace them with the notice
   and other provisions required by the GPL or the LGPL. If you do not delete
   the provisions above, a recipient may use your version of this file under
   the terms of any one of the MPL, the GPL or the LGPL.

   ***** END LICENSE BLOCK *****
*/

import java.security.MessageDigest;
import java.lang.reflect.Array;
```

```
public class Example2b {

  // Your keys here
  private final static long p1 = 0xEC6CC589AEFD1E75L;
  private final static long p2 = 0xFCEC0A6A82747B3FL;

  private static int error;

  //The relative path to uskjni.dll from your main class.
  private final static String apiJniRelativePath = ".";

  // SmartDongle JNI library.
  private final static String apiJniName = "libuskjni.so";

  // Some application data
  private final static String data = "This is data written to the SmartDongle, it can
be serial numbers, program parameters, etc.  Note that this data can be read by
observing activity on the USB bus. Refer to Example3a and Example3b for example
 of encrypting your application data.";


  public static void main(String args[]) {


    /**
       The code below will be inserted into your main class
    */


    String fileSeparator = System.getProperty("file.separator");
    String apiJniAbsolutePath =
      System.getProperty("user.dir") +
      fileSeparator +
      apiJniRelativePath +
      fileSeparator +
      apiJniName;

    if (!smartDongleInitialize(apiJniAbsolutePath)) {
      // FAIL exiting...
      return;
    }

    // Read application data
    byte[] dataBytes = data.getBytes();
    int address = 0;
    int dataLength = Array.getLength(dataBytes);
    byte inBytes[] = new byte[dataLength];
    error = smartDongleRead(p1,
```

```
                        p2,
                        address,
                        inBytes,
                        dataLength);
    if (error != 0) {
       // FAIL exiting...
       return;
    }

    /*
      Note: This exercise simply compares the data read from the SmartDongle
      with the expected application data and returns before your main classcode
      is executed, a developer could do any number of things with the data here.
    */
    int i;
    for (i = 0; i < dataLength; i++) {
      if (inBytes[i] != dataBytes[i]) {
        // FAIL exiting...
        return;
      }
    }



    /*

      Your main class code goes here

    */
    System.out.println("Success: Will execute main class");



  }


  /*
   *  SmartDongle native methods here
   */
  private static native int smartDongleRead(long p1,
                                    long p2,
                                    int addr,
                                    byte[] data,
                                    int size);

  private static boolean smartDongleInitialize(String apiJniPath) {

// AWK_USKJNI_SIG_BEGIN
byte sig[] = {-85, -55, -2, 64, -38, -124, -47, 64, -105, 121, -8, -4, 84, -116,
19, -128, -128, 120, 44, -5, };
```

```
// AWK_USKJNI_SIG_END

    MessageDigest md = null;

    /*
     * Check uskjni SHA fingerprint
     */
    try {
      md = MessageDigest.getInstance("SHA");

      java.io.FileInputStream in = new java.io.FileInputStream(apiJniPath);
      int len;

      byte[] input = new byte[512];

      while ((len = in.read(input)) > 0) {
        md.update(input, 0, len);
      }

      byte[] digest = md.digest();

      // should be 20 bytes, 160 bits long
      if (digest.length != sig.length) {
        return false;
      }

      // Check uskjni.dll SHA fingerprint
      for (int i = 0; i<digest.length; i++) {
        if (digest[i] != sig[i]) {
          return false;
        }
      }

      System.load(apiJniPath);
    }
    catch (java.io.IOException ioe) {
      System.out.println("Caught Exception:  " + ioe);
    }
    catch (java.security.NoSuchAlgorithmException nsae) {
      System.out.println("Caught Exception:  " + nsae);
    }

    return true;
  }
}
```

If you store data on a SmartDongle you may wish to encrypt the data transmitted over the USB bus.

Examples 3 below use AES encryption. A developer could use example3a.c as a utility to write

passwords, serial numbers, etc to a SmartDongle that can be read back and used in an application as demonstrated by example3b.c

4. Example 3a: Utility to encrypt a user string using AES and write this string to a SmartDongle

```
/**
   Example3a.java  Write then verify an encrypted secret to a SmartDongle.

   MicroWorks, Inc.
   2808 North Cole Road
   Boise, ID 83704

   www.smartdongle.com


   ***** BEGIN LICENSE BLOCK *****

   Version: MPL 1.1/GPL 2.0/LGPL 2.1

   The contents of this file are subject to the Mozilla Public License
   Version 1.1 (the “License”); you may not use this file except in
   compliance with the License. You may obtain a copy of the License at
   http://www.mozilla.org/MPL/

   Software distributed under the License is distributed on an “AS IS” basis,
   WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License
   for the specific language governing rights and limitations under the
   License.

   The Original Code is MicroWorks, Inc. code.

   The Initial Developer of the Original Code is MicroWorks, Inc. Portions
   created by MicroWorks, Inc. are Copyright (C) 2008
   MicroWorks, Inc. All Rights Reserved.

   Contributor(s):

   Alternatively, the contents of this file may be used under the terms of
   either the GNU General Public License Version 2 or later (the “GPL”), or
   the GNU Lesser General Public License Version 2.1 or later (the “LGPL”),
   in which case the provisions of the GPL or the LGPL are applicable instead
   of those above. If you wish to allow use of your version of this file only
   under the terms of either the GPL or the LGPL, and not to allow others to
   use your version of this file under the terms of the MPL, indicate your
   decision by deleting the provisions above and replace them with the notice
   and other provisions required by the GPL or the LGPL. If you do not delete
   the provisions above, a recipient may use your version of this file under
```

```
    the terms of any one of the MPL, the GPL or the LGPL.

    ***** END LICENSE BLOCK *****
*/

import java.lang.reflect.Array;
import java.security.MessageDigest;
import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;


public class Example3a {

  private static int error;

  //The relative path to uskjni.dll from your main class.
  private final static String apiJniRelativePath = ".";

  // Your keys here
  private final static long p1 = 0xEC6CC589AEFD1E75L;
  private final static long p2 = 0xFCEC0A6A82747B3FL;

  /*
    This is a demo 128 bit encryption key,
    chose some another 16 character key for your project.
  */
  private final static byte[] AESkey = {
    '7',  'o',  'b',  'd',
    'g',  'E',  'n',  '2',
    '\'', 'T',  'H',  '>',
    ']',  'S',  'E',  '|'};

  // SmartDongle JNI library.
  private final static String apiJniName = "libuskjni.so";

  // Encrypted data stored on SmartDongle.
  private final static String plainText = "This is the secret written to SmartDongle,
 it can be passwords, serial numbers, program parameters, etc";

  private final static int aesBlockSize = 128;

  public static void main(String args[]) {

    String fileSeparator = System.getProperty("file.separator");
    String apiJniAbsolutePath =
      System.getProperty("user.dir") +
      fileSeparator +
      apiJniRelativePath +
```

```
  fileSeparator +
  apiJniName;

if (!smartDongleInitialize(apiJniAbsolutePath)) {
  System.out.println("SmartDongle JNI initialisation failed");
  return;
}


try {

  /*
    This instantiates a AES key from the user's key bytes.
    The key is used to initialize a AES cipher for
    encryption and decryption.
  */
  SecretKeySpec skeySpec = new SecretKeySpec(AESkey, "AES");
  Cipher cipher = Cipher.getInstance("AES");
  cipher.init(Cipher.ENCRYPT_MODE, skeySpec);

  // Encrypt the user's application data
  byte[] cipherText = cipher.doFinal(plainText.getBytes());
  int cipherTextLength = Array.getLength(cipherText);

  // Write encrypted data to SmartDongle
  int address = 0;
  error = smartDongleWrite(p1,
                           p2,
                           address,
                           cipherText,
                           cipherTextLength);
  if (error != 0) {
    System.out.println(
      "Error writing message: " + error + " exiting...");
    return;
  }

  // Verify data
  byte[] input = new byte[cipherTextLength];
  error = smartDongleRead(p1,
                          p2,
                          address,
                          input,
                          cipherTextLength);

  int i;
  for (i = 0; i < cipherTextLength; i++) {
    if (input[i] != cipherText[i]) {
      System.out.println("Failed to write encrypted user data, mismatch");
      return;
```

```
      }
    }

    System.out.println("Success writing encrypted user data of length "
                       + cipherTextLength);

    // Write AES key to SmartDongle
    error = smartDongleWrite(p1,
                             p2,
                             address + cipherTextLength,
                             AESkey,
                             aesBlockSize / 8);  // block size is in bits
    if (error != 0) {
      System.out.println("Error AES key: " + error + " exiting...");
      return;
    }

    // Verify that key was written
    error = smartDongleRead(p1,
                            p2,
                            address + cipherTextLength,
                            input,
                            aesBlockSize / 8);  // block size is in bits

    for (i = 0; i < aesBlockSize / 8; i++) {
      if (input[i] != AESkey[i]) {
        System.out.println("Failed to write user AES key, mismatch");
        return;
      }
    }

    System.out.println("Success writing AES key of length " +
                       Array.getLength(AESkey));
  }
  catch (java.security.NoSuchAlgorithmException nsafe) {
    System.out.println("Caught Exception:  " + nsafe);
  }
  catch (javax.crypto.NoSuchPaddingException nspe) {
    System.out.println("Caught Exception:  " + nspe);
  }
  catch (javax.crypto.BadPaddingException bpe) {
    System.out.println("Caught Exception:  " + bpe);
  }
  catch (javax.crypto.IllegalBlockSizeException ibse) {
    System.out.println("Caught Exception:  " + ibse);
  }
  catch (java.security.InvalidKeyException ike) {
    System.out.println("Caught Exception:  " + ike);
  }
}
```

```
  /*
   *  SmartDongle native methods here
   */
  private static native int smartDongleRead(long p1,
                                            long p2,
                                            int addr,
                                            byte[] data,
                                            int size);
  private static native int smartDongleWrite(long p1,
                                             long p2,
                                             int addr,
                                             byte[] data,
                                             int size);

  private static boolean smartDongleInitialize(String apiJniPath) {

// AWK_USKJNI_SIG_BEGIN
byte sig[] = {96, 17, -114, -27, -75, 14, -53, -127, -18, 119, -28, -90, -20, -123,
-12, -25, -66, -72, -55, 73, };
// AWK_USKJNI_SIG_END

    MessageDigest md = null;

    /*
     * Check uskjni SHA fingerprint
     */
    try {
      md = MessageDigest.getInstance("SHA");

      java.io.FileInputStream in = new java.io.FileInputStream(apiJniPath);
      int len;

      byte[] input = new byte[512];

      while ((len = in.read(input)) > 0) {
        md.update(input, 0, len);
      }

      byte[] digest = md.digest();

      // should be 20 bytes, 160 bits long
      if (digest.length != sig.length) {
        return false;
      }

      // Check uskjni.dll SHA fingerprint
      for (int i = 0; i<digest.length; i++) {
        if (digest[i] != sig[i]) {
```

```
        return false;
      }
    }

    System.load(apiJniPath);
  }
  catch (java.io.IOException ioe) {
    System.out.println("Caught Exception:  " + ioe);
  }
  catch (java.security.NoSuchAlgorithmException nsae) {
    System.out.println("Caught Exception:  " + nsae);
  }


  return true;
 }
}
```

5. Example 3b: An application that reads and decrypt the secret written in Example3a

```
/**
   Example3b.java  Verify a secret on a SmartDongle written using the utility
                   in Example3a

   MicroWorks, Inc.
   2808 North Cole Road
   Boise, ID 83704

   www.smartdongle.com


   ***** BEGIN LICENSE BLOCK *****

   Version: MPL 1.1/GPL 2.0/LGPL 2.1

   The contents of this file are subject to the Mozilla Public License
   Version 1.1 (the "License"); you may not use this file except in
   compliance with the License. You may obtain a copy of the License at
   http://www.mozilla.org/MPL/

   Software distributed under the License is distributed on an "AS IS" basis,
   WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License
   for the specific language governing rights and limitations under the
   License.

   The Original Code is MicroWorks, Inc. code.

   The Initial Developer of the Original Code is MicroWorks, Inc. Portions
```

*/

```
import java.lang.reflect.Array;
import java.security.MessageDigest;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;


public class Example3b {

  private static int error;

  //The relative path to uskjni.dll from your main class.
  private final static String apiJniRelativePath = ".";

  // Your keys here
  private final static long p1 = 0xEC6CC589AEFD1E75L;
  private final static long p2 = 0xFCEC0A6A82747B3FL;

  // SmartDongle JNI library.
  private final static String apiJniName = "libuskjni.so";

  // Encrypted data stored on SmartDongle.
  private final static String plainText = "This is the secret written to SmartDongle,
 it can be passwords, serial numbers, program parameters, etc";

  private final static int aesBlockSize = 128;


  public static void main(String args[]) {
```

```
String fileSeparator = System.getProperty("file.separator");
String apiJniAbsolutePath =
  System.getProperty("user.dir") +
  fileSeparator +
  apiJniRelativePath +
  fileSeparator +
  apiJniName;

if (!smartDongleInitialize(apiJniAbsolutePath)) {
  // FAIL exiting...
  return;
}

/*
    First, calculate the number of bytes you will need to read from the
    SmartDongle since the encrypted bytes are rounded up to the nearest
    multiple AES block size.
*/
int plainTextLength = plainText.length();

int cipherTextBlocks = plainTextLength / (aesBlockSize / 8);
if (plainTextLength % (aesBlockSize / 8) != 0) {
    cipherTextBlocks++;
}

// Total cipher length to be read from SmartDongle.
int cipherTextLength = cipherTextBlocks * (aesBlockSize / 8);


try {

  /*
    This instantiates an AES key read from SmartDongle written in
    Example3a. This key is used to initialize an AES cipher for decryption.

    Note that you could simply store the key in this class as
    private final static byte[] AESkey = {
      'y','o','u','r',' ','1','6', ' ','b','y','t','e',' ','k','e','y'}

    Given the ease of which Java is decompliled, storing the key outside
    of the application is a good ideas since this makes for an extra level
    of obfuscation.
  */

  // Retrieve the AES key from the SmartDongle
  byte[] AESkey = new byte[aesBlockSize / 8];
  int address = 0;
  error = smartDongleRead(p1,
                          p2,
```

```
                    address + cipherTextLength,
                    AESkey,
                    aesBlockSize / 8);
  if (error != 0) {
    // FAIL exiting...
    return;
  }

  /*
     This instantiates a AES key retrieved from the SmartDongle
     key is used to initialize a AES cipher for decryption.
  */
  SecretKeySpec skeySpec = new SecretKeySpec(AESkey, “AES”);
  Cipher cipher = Cipher.getInstance(“AES”);
  cipher.init(Cipher.DECRYPT_MODE, skeySpec);


  // Retrieve users encrypted application data
  byte[] input = new byte[cipherTextLength];
  error = smartDongleRead(p1,
                    p2,
                    address,
                    input,
                    cipherTextLength);
  if (error != 0) {
    // FAIL exiting...
    return;
  }

  // Decrypt the user’s application data
  byte[] decryptedBytes = cipher.doFinal(input);


  /*
    Verify the applicaton data that was written in Example3a

    Note: This is just an excercise, a developer could do any number
    of things with the application data here.
  */
  int i;
  byte[] plainTextBytes = plainText.getBytes();
  for (i = 0; i < plainText.length(); i++) {
    if (decryptedBytes[i] != plainTextBytes[i] ) {
      // FAIL exiting...
      return;
    }
  }


}
```

```
  catch (java.security.NoSuchAlgorithmException nsafe) {
    System.out.println("Caught Exception:  " + nsafe);
  }
  catch (javax.crypto.NoSuchPaddingException nspe) {
    System.out.println("Caught Exception:  " + nspe);
  }
  catch (javax.crypto.BadPaddingException bpe) {
    System.out.println("Caught Exception:  " + bpe);
  }
  catch (javax.crypto.IllegalBlockSizeException ibse) {
    System.out.println("Caught Exception:  " + ibse);
  }
  catch (java.security.InvalidKeyException ike) {
    System.out.println("Caught Exception:  " + ike);
  }



  /*

     Your main class code goes here

  */
  System.out.println("Success: Will execute main class");




}


/*
 *  SmartDongle native methods here
 */
private static native int smartDongleRead(long p1,
                                          long p2,
                                          int addr,
                                          byte[] data,
                                          int size);
private static native int smartDongleWrite(long p1,
                                           long p2,
                                           int addr,
                                           byte[] data,
                                           int size);

private static boolean smartDongleInitialize(String apiJniPath) {

// AWK_USKJNI_SIG_BEGIN
```

```
byte sig[] = {-1, -50, -6, 76, 5, 42, 17, -41, 48, -34, 57, -88, -62, -88, -82, -48,
-39, 91, 78, 127, };
// AWK_USKJNI_SIG_END

    MessageDigest md = null;

    /*
     * Check uskjni SHA fingerprint
     */
    try {
      md = MessageDigest.getInstance("SHA");

      java.io.FileInputStream in = new java.io.FileInputStream(apiJniPath);
      int len;

      byte[] input = new byte[512];

      while ((len = in.read(input)) > 0) {
        md.update(input, 0, len);
      }

      byte[] digest = md.digest();

      // should be 20 bytes, 160 bits long
      if (digest.length != sig.length) {
        return false;
      }

      // Check uskjni.dll SHA fingerprint
      for (int i = 0; i<digest.length; i++) {
        if (digest[i] != sig[i]) {
          return false;
        }
      }

      System.load(apiJniPath);
    }
    catch (java.io.IOException ioe) {
      System.out.println("Caught Exception:  " + ioe);
    }
    catch (java.security.NoSuchAlgorithmException nsae) {
      System.out.println("Caught Exception:  " + nsae);
    }


    return true;
  }
}
```

Example 4 below takes advantage of Java Object Serialization.

6. Example 4a: Utility to initialize a user's Properties Object, serialize the object, encrypt the serial stream, then write this stream to a SmartDongle

```
/**
   Example4a.java  Utility that will write a properties list to a SmartDongle.
                   The properties list will be read by an example application
                   program, Example4b.java



   The map of the data stored to SmartDongle:

         16 bytes              2 bytes                 n bytes
   | Customer's AES key | Property Object size |    Customer's Property    |
    ------------------   --------------------  ---------------------------
    0     field 1     15 16      field 2     17 18         field 3         n


   MicroWorks, Inc.
   2808 North Cole Road
   Boise, ID 83704

   www.smartdongle.com


   ***** BEGIN LICENSE BLOCK *****

   Version: MPL 1.1/GPL 2.0/LGPL 2.1

   The contents of this file are subject to the Mozilla Public License
   Version 1.1 (the "License"); you may not use this file except in
   compliance with the License. You may obtain a copy of the License at
   http://www.mozilla.org/MPL/

   Software distributed under the License is distributed on an "AS IS" basis,
   WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License
   for the specific language governing rights and limitations under the
   License.

   The Original Code is MicroWorks, Inc. code.

   The Initial Developer of the Original Code is MicroWorks, Inc. Portions
   created by MicroWorks, Inc. are Copyright (C) 2008
   MicroWorks, Inc. All Rights Reserved.

   Contributor(s):

   Alternatively, the contents of this file may be used under the terms of
```

```
   either the GNU General Public License Version 2 or later (the “GPL”), or
   the GNU Lesser General Public License Version 2.1 or later (the “LGPL”),
   in which case the provisions of the GPL or the LGPL are applicable instead
   of those above. If you wish to allow use of your version of this file only
   under the terms of either the GPL or the LGPL, and not to allow others to
   use your version of this file under the terms of the MPL, indicate your
   decision by deleting the provisions above and replace them with the notice
   and other provisions required by the GPL or the LGPL. If you do not delete
   the provisions above, a recipient may use your version of this file under
   the terms of any one of the MPL, the GPL or the LGPL.

   ***** END LICENSE BLOCK *****
*/

import java.lang.reflect.Array;
import java.io.ByteArrayOutputStream;
import java.security.MessageDigest;
import java.util.Properties;
import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;


/*
   The map of the data stored to SmartDongle:

         16 bytes              2 bytes                  n bytes
   | Customer’s AES key | Property Object size |    Customer’s Property    |
    ------------------   --------------------  --------------------------
    0     field 1     15 16      field 2     17 18         field 3         n

*/

public class Example4a {

  private static int error;

  //The relative path to uskjni.dll from your main class.
  private final static String apiJniRelativePath = “.”;

  // Your keys here
  private final static long p1 = 0xEC6CC589AEFD1E75L;
  private final static long p2 = 0xFCEC0A6A82747B3FL;

  /*
    This is a demo 128 bit encryption key,
    chose some another 16 character key for your project.
  */
  private final static byte[] AESkey = {
    ‘7’,  ‘o’,  ‘b’,  ‘d’,
```

```
  'g',  'E',  'n',  '2',
  '\'', 'T',  'H',  '>',
  ']',  'S',  'E',  '|'};


// Parameters needed to store Properties object to SmartDongle.
private final static int aesBlockSize = 128;
private final static int keyAddress = 0;
private final static int propertiesSizeAddress = aesBlockSize / 8;
private final static int propertiesObjectAddress = propertiesSizeAddress + 2;


// SmartDongle JNI library.
private final static String apiJniName = "libuskjni.so";


public static void main(String args[]) {


  /**
   * Initialise your Properties object that will be saved to SmartDongle.
   */
  Properties properties = new Properties();
  properties.setProperty("customerName", "Widget City, Inc.");
  properties.setProperty("licenseSN", "1111-2222-3333-4444-5555-6666");
  properties.setProperty("licenseExpiration", "30 May 2010");
  properties.setProperty("usageCurrent", "0");
  properties.setProperty("usageLimit", "0");

  // Properties comment.
  String propertiesHeader =
    new String("Example property list stored on a SmartDongle");


  /*
      Initialize SmartDongle interface
  */
  String fileSeparator = System.getProperty("file.separator");
  String apiJniAbsolutePath =
    System.getProperty("user.dir") +
    fileSeparator +
    apiJniRelativePath +
    fileSeparator +
    apiJniName;

  if (!smartDongleInitialize(apiJniAbsolutePath)) {
    System.out.println("SmartDongle JNI initialisation failed");
    return;
  }
```

```
/*
   This code will process and save your Properties to a SmartDongle

   1) Instantiate an AES key from the user's key bytes.
   2) Initialize an AES cipher for encryption.
   3) Write the user's Properties into an output byte stream.
   4) Encrypt user's Properties byte stream.
   5) Save user's raw key bytes to SmartDongle
   6) Save length of user's encrypted Properties byte stream to Smartdongle
   7) Save encrypted Properties byte stream to SmartDongle.
*/
int cipherTextLength = 0;
try {

  int i;

  /*

    Instantiate an AES key from the user's key bytes.
    Initialize an AES cipher for encryption.

  */
  SecretKeySpec skeySpec = new SecretKeySpec(AESkey, "AES");
  Cipher cipher = Cipher.getInstance("AES");
  cipher.init(Cipher.ENCRYPT_MODE, skeySpec);



  /*
    Write the user's Properties into an output byte stream.
    Encrypt user's Properties byte stream.
  */
  ByteArrayOutputStream baos = new ByteArrayOutputStream();
  properties.store(baos, propertiesHeader);

  // Convert output byte stream into a byte array
  byte[] propertiesByteArray = baos.toByteArray();

  // Encrypt the converted properties list
  byte[] cipherText = cipher.doFinal(propertiesByteArray);
  cipherTextLength = Array.getLength(cipherText);



  /*
    Write user's raw key bytes to SmartDongle
  */
  error = smartDongleWrite(p1,
```

```
                    p2,
                    keyAddress,
                    AESkey,
                    aesBlockSize / 8);  // block size is in bits
if (error != 0) {
  System.out.println("Error writing user's key bytes: " + error);
  return;
}

// Verify that key was written
byte[] field1 = new byte[aesBlockSize / 8];
error = smartDongleRead(p1,
                    p2,
                    keyAddress,
                    field1,
                    aesBlockSize / 8);  // block size is in bits
for (i = 0; i < aesBlockSize / 8; i++) {
  if (AESkey[i] != field1[i]) {
    System.out.println("Failed to write user's key bytes: mismatch");
    return;
  }
}

System.out.println("Success writing AES key of length " +
                Array.getLength(AESkey));



/*
  Save length of user's encrypted Properties byte stream to Smartdongle.
*/

// We need to record the properties list cipher length
Short s0 = new Short((short)(cipherTextLength & 0xff));
Short s1 = new Short((short)((cipherTextLength & 0xff00) >> 8));

// Convert the shorts above into a byte array
byte[] propertiesSize = new byte[2];
propertiesSize[0] = s0.byteValue();
propertiesSize[1] = s1.byteValue();

// Write out the byte array with length of properties list
error = smartDongleWrite(p1,
                    p2,
                    propertiesSizeAddress,
                    propertiesSize,
                    2);
if (error != 0) {
  System.out.println("Error writing properties length field: " +
                error);
```

```
  return;
}


// Verify that properties list length was written
byte[] field2 = new byte[2];
error = smartDongleRead(p1,
                        p2,
                        propertiesSizeAddress,
                        field2,
                        2);

for (i = 0; i < 2; i++) {
  if (propertiesSize[i] != field2[i]) {
    System.out.println("Failed to write length field: mismatch");
    return;
  }
}


System.out.println("Success writing properties list length field ");



/*
  Save encrypted Properties byte stream to SmartDongle.
*/
error = smartDongleWrite(p1,
                         p2,
                         propertiesObjectAddress,
                         cipherText,
                         cipherTextLength);
if (error != 0) {
  System.out.println("Error writing properties field: " +
                     error);
  return;
}

// Verify that properties list was written
byte[] field3 = new byte[cipherTextLength];
error = smartDongleRead(p1,
                        p2,
                        propertiesObjectAddress,
                        field3,
                        cipherTextLength);
for (i = 0; i < cipherTextLength; i++) {
  if (cipherText[i] != field3[i]) {
    System.out.println("Failed to write properties field: mismatch");
    return;
  }
```

```
    }

  }
  catch (java.io.IOException ioe) {
    System.out.println("Caught Exception:  " + ioe);
  }
  catch (java.security.NoSuchAlgorithmException nsafe) {
    System.out.println("Caught Exception:  " + nsafe);
  }
  catch (javax.crypto.NoSuchPaddingException nspe) {
    System.out.println("Caught Exception:  " + nspe);
  }
  catch (javax.crypto.BadPaddingException bpe) {
    System.out.println("Caught Exception:  " + bpe);
  }
  catch (javax.crypto.IllegalBlockSizeException ibse) {
    System.out.println("Caught Exception:  " + ibse);
  }
  catch (java.security.InvalidKeyException ike) {
    System.out.println("Caught Exception:  " + ike);
  }


  // Get serial number for report
  byte[] smartDongleSerialNumber = new byte[14];
  smartDongleGetSerialNumber(smartDongleSerialNumber);

  System.out.println("Finished writing property list of length "
                     + cipherTextLength
                     + " to sn:"
                     + new String(smartDongleSerialNumber));
}


/*
 *  SmartDongle native methods here
 */
private static native int smartDongleRead(long p1,
                                          long p2,
                                          int addr,
                                          byte[] data,
                                          int size);
private static native int smartDongleWrite(long p1,
                                           long p2,
                                           int addr,
                                           byte[] data,
                                           int size);

private static native int smartDongleGetSerialNumber(byte[] sn);
```

```
  private static boolean smartDongleInitialize(String apiJniPath) {

// AWK_USKJNI_SIG_BEGIN
byte sig[] = {98, -15, -97, 1, 5, 44, 52, 105, 116, 71, 6, -18, -75, 22, -113, 12,
29, -7, 122, -78, };
// AWK_USKJNI_SIG_END

    MessageDigest md = null;

    /*
     * Check uskjni SHA fingerprint
     */
    try {
      md = MessageDigest.getInstance("SHA");

      java.io.FileInputStream in = new java.io.FileInputStream(apiJniPath);
      int len;

      byte[] input = new byte[512];

      while ((len = in.read(input)) > 0) {
        md.update(input, 0, len);
      }

      byte[] digest = md.digest();

      // should be 20 bytes, 160 bits long
      if (digest.length != sig.length) {
        return false;
      }

      // Check uskjni.dll SHA fingerprint
      for (int i = 0; i<digest.length; i++) {
        if (digest[i] != sig[i]) {
          return false;
        }
      }

      System.load(apiJniPath);
    }
    catch (java.io.IOException ioe) {
      System.out.println("Caught Exception:  " + ioe);
    }
    catch (java.security.NoSuchAlgorithmException nsae) {
      System.out.println("Caught Exception:  " + nsae);
    }


    return true;
  }
```

}

7. Example 4b: An application that retrieves the user's Properties Object that was serialized and encrypted in Example 4a'

```
/**
   Example4b.java

   Retrieve a user's properties object written with example3a.java utility.


   The map of the data stored to SmartDongle:

         16 bytes              2 bytes                  n bytes
   | Customer's AES key | Property Object size |    Customer's Property    |
    ------------------   --------------------  --------------------------
    0     field 1     15 16      field 2     17 18         field 3         n


   MicroWorks, Inc.
   2808 North Cole Road
   Boise, ID 83704

   www.smartdongle.com


   ***** BEGIN LICENSE BLOCK *****

   Version: MPL 1.1/GPL 2.0/LGPL 2.1

   The contents of this file are subject to the Mozilla Public License
   Version 1.1 (the "License"); you may not use this file except in
   compliance with the License. You may obtain a copy of the License at
   http://www.mozilla.org/MPL/

   Software distributed under the License is distributed on an "AS IS" basis,
   WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License
   for the specific language governing rights and limitations under the
   License.

   The Original Code is MicroWorks, Inc. code.

   The Initial Developer of the Original Code is MicroWorks, Inc. Portions
   created by MicroWorks, Inc. are Copyright (C) 2008
   MicroWorks, Inc. All Rights Reserved.

   Contributor(s):

   Alternatively, the contents of this file may be used under the terms of
   either the GNU General Public License Version 2 or later (the "GPL"), or
```

```
    the GNU Lesser General Public License Version 2.1 or later (the “LGPL”),
    in which case the provisions of the GPL or the LGPL are applicable instead
    of those above. If you wish to allow use of your version of this file only
    under the terms of either the GPL or the LGPL, and not to allow others to
    use your version of this file under the terms of the MPL, indicate your
    decision by deleting the provisions above and replace them with the notice
    and other provisions required by the GPL or the LGPL. If you do not delete
    the provisions above, a recipient may use your version of this file under
    the terms of any one of the MPL, the GPL or the LGPL.

    ***** END LICENSE BLOCK *****
*/

import java.lang.reflect.Array;
import java.io.ByteArrayInputStream;
import java.security.MessageDigest;
import java.util.Properties;
import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;


/*
    The map of the data stored to SmartDongle:

          16 bytes              2 bytes                  n bytes
    | Customer’s AES key | Property Object size |    Customer’s Property    |
     ------------------   --------------------  ---------------------------
     0     field 1     15 16      field 2     17 18         field 3         n

*/

public class Example4b {

  private static int error;

  // Your keys here
  private final static long p1 = 0xEC6CC589AEFD1E75L;
  private final static long p2 = 0xFCEC0A6A82747B3FL;

  // Your properties to be retrieved from a SmartDongle
  private static Properties properties;

  // Parameters needed to retrieve Properties object from SmartDongle.
  private final static int aesBlockSize = 128;
  private final static int keyAddress = 0;
  private final static int propertiesSizeAddress = aesBlockSize / 8;
  private final static int propertiesObjectAddress = propertiesSizeAddress + 2;

  //The relative path to uskjni.dll from your main class.
```

```
private final static String apiJniRelativePath = ".";

// SmartDongle JNI library.
private final static String apiJniName = "libuskjni.so";


public static void main(String args[]) {

 /*
    Initialize SmartDongle interface
 */
 String fileSeparator = System.getProperty("file.separator");
 String apiJniAbsolutePath =
  System.getProperty("user.dir") +
  fileSeparator +
  apiJniRelativePath +
  fileSeparator +
  apiJniName;

 if (!smartDongleInitialize(apiJniAbsolutePath)) {
  // FAIL exiting...
  return;
 }



 /*
    This code will retrieve your Properties object from a SmartDongle

    1) Read user's raw key bytes from Smartdongle.
    2) Instantiate an AES key from the user's key bytes.
    3) Initialize an AES cipher for encryption.
    4) Read the encrypted input stream length from Smartdongle.
    5) Read the encrypted input stream from SmartDongle.
    6) Decrypt the input stream.
    7) Load the user's Properties from the decrypted input stream.
 */
 int cipherTextLength = 0;
 try {

  /*
    1) Read user's raw key bytes from Smartdongle.
  */
  byte[] field1 = new byte[aesBlockSize / 8];
  error = smartDongleRead(p1,
                          p2,
                          keyAddress,
                          field1,
                          aesBlockSize / 8);  // block size is in bits
  if (error != 0) {
```

```
  // FAIL exiting...
  return;
}



/*
  2) Instantiate an AES key from the user's key bytes.
  3) Initialize an AES cipher for encryption.
*/
SecretKeySpec skeySpec = new SecretKeySpec(field1, "AES");
Cipher cipher = Cipher.getInstance("AES");
cipher.init(Cipher.DECRYPT_MODE, skeySpec);



/*
  4) Read the encrypted input stream length from Smartdongle.
*/
byte[] field2 = new byte[2];
error = smartDongleRead(p1,
                        p2,
                        propertiesSizeAddress,
                        field2,
                        2);
if (error != 0) {
  // FAIL exiting...
  return;
}

/*
  The byte array length was written as unsigned data,
  and the serialised properties length must be <= 32767. (user memory)
*/
int field3Length =
  (field2[1] >= 0 ? field2[1] : 256 + field2[1]) << 8;
field3Length += field2[0] >= 0 ? field2[0] : 256 + field2[0];



/*
  5) Read the encrypted input stream from SmartDongle.
*/
byte[] field3 = new byte[field3Length];
error = smartDongleRead(p1,
                        p2,
                        propertiesObjectAddress,
                        field3,
                        field3Length);
if (error != 0) {
```

```
    // FAIL exiting...
    return;
  }



  /*
    6) Decrypt the input stream.
  */
  byte[] decipheredStream = cipher.doFinal(field3);



  /*
    7) Load the user's Properties from the decrypted input stream.
  */
  properties = new Properties();
  ByteArrayInputStream bais = new ByteArrayInputStream(decipheredStream);
  properties.load(bais);

}
catch (java.io.IOException ioe) {
  System.out.println("Caught Exception:  " + ioe);
}
catch (java.security.NoSuchAlgorithmException nsafe) {
  System.out.println("Caught Exception:  " + nsafe);
}
catch (javax.crypto.NoSuchPaddingException nspe) {
  System.out.println("Caught Exception:  " + nspe);
}
catch (javax.crypto.BadPaddingException bpe) {
  System.out.println("Caught Exception:  " + bpe);
}
catch (javax.crypto.IllegalBlockSizeException ibse) {
  System.out.println("Caught Exception:  " + ibse);
}
catch (java.security.InvalidKeyException ike) {
  System.out.println("Caught Exception:  " + ike);
}



/*
   Your main class code goes here
*/

System.out.println("Success: Your application can use these properties:");

System.out.println("    Customer Name: " +
                   properties.getProperty("customerName"));
```

```
    System.out.println("    Serial Number: " +
                       properties.getProperty("licenseSN"));

    System.out.println("    Expiration Date: " +
                       properties.getProperty("licenseExpiration"));

    System.out.println("    Usage: " +
                       properties.getProperty("usageCurrent"));

    System.out.println("    Usage Limit: " +
                       properties.getProperty("usageLimit"));
  }


  /*
   *  SmartDongle native methods here
   */
  private static native int smartDongleRead(long p1,
                                            long p2,
                                            int addr,
                                            byte[] data,
                                            int size);
  private static native int smartDongleWrite(long p1,
                                             long p2,
                                             int addr,
                                             byte[] data,
                                             int size);

  private static native int smartDongleGetSerialNumber(byte[] sn);

  private static boolean smartDongleInitialize(String apiJniPath) {

// AWK_USKJNI_SIG_BEGIN
byte sig[] = {48, 77, -96, 80, 75, 65, -107, 44, 106, 61, 123, -54, 105, 106, 82,
-84, -14, 72, 11, -70, };
// AWK_USKJNI_SIG_END

    MessageDigest md = null;

    /*
     * Check uskjni SHA fingerprint
     */
    try {
      md = MessageDigest.getInstance("SHA");

      java.io.FileInputStream in = new java.io.FileInputStream(apiJniPath);
      int len;

      byte[] input = new byte[512];
```

```
      while ((len = in.read(input)) > 0) {
        md.update(input, 0, len);
      }

      byte[] digest = md.digest();

      // should be 20 bytes, 160 bits long
      if (digest.length != sig.length) {
        return false;
      }

      // Check uskjni.dll SHA fingerprint
      for (int i = 0; i<digest.length; i++) {
        if (digest[i] != sig[i]) {
          return false;
        }
      }

      System.load(apiJniPath);
    }
    catch (java.io.IOException ioe) {
      System.out.println("Caught Exception:  " + ioe);
    }
    catch (java.security.NoSuchAlgorithmException nsae) {
      System.out.println("Caught Exception:  " + nsae);
    }


    return true;
  }
}
```

8. Read SmartDongle serial number

```
/**
    Example5.java  Return a SmartDongle serial number


    MicroWorks, Inc.
    2808 North Cole Road
    Boise, ID 83704

    www.smartdongle.com


    ***** BEGIN LICENSE BLOCK *****

    Version: MPL 1.1/GPL 2.0/LGPL 2.1
```

*/

```
import java.security.MessageDigest;

public class Example5 {

  private static int error;

  //The relative path to uskjni.dll from your main class.
  private final static String apiJniRelativePath = “.”;

  // SmartDongle JNI library.
  private final static String apiJniName = “libuskjni.so”;


  public static void main(String args[]) {


    /*
```

```
      Initialize SmartDongle interface
    */
    String fileSeparator = System.getProperty("file.separator");
    String apiJniAbsolutePath =
      System.getProperty("user.dir") +
      fileSeparator +
      apiJniRelativePath +
      fileSeparator +
      apiJniName;

    if (!smartDongleInitialize(apiJniAbsolutePath)) {
      System.out.println("SmartDongle JNI initialisation failed");
      return;
    }


    // Get serial number for report
    byte[] smartDongleSerialNumber = new byte[14];
    smartDongleGetSerialNumber(smartDongleSerialNumber);

    System.out.println("SmartDongle serial number: "
                    + new String(smartDongleSerialNumber));
  }


  /*
   *  SmartDongle native methods here
   */

  private static native int smartDongleGetSerialNumber(byte[] sn);

  private static boolean smartDongleInitialize(String apiJniPath) {

// AWK_USKJNI_SIG_BEGIN
byte sig[] = {67, -128, 46, 36, 124, -27, -96, -62, -73, 43, 22, -127, -15, -118, -62, 98,
-116, 69, -6, -58, };
// AWK_USKJNI_SIG_END

    MessageDigest md = null;

    /*
     * Check uskjni SHA fingerprint
     */
    try {
      md = MessageDigest.getInstance("SHA");

      java.io.FileInputStream in = new java.io.FileInputStream(apiJniPath);
      int len;

      byte[] input = new byte[512];
```

```
      while ((len = in.read(input)) > 0) {
        md.update(input, 0, len);
      }

      byte[] digest = md.digest();

      // should be 20 bytes, 160 bits long
      if (digest.length != sig.length) {
        return false;
      }

      // Check uskjni.dll SHA fingerprint
      for (int i = 0; i<digest.length; i++) {
        if (digest[i] != sig[i]) {
          return false;
        }
      }

      System.load(apiJniPath);
    }
    catch (java.io.IOException ioe) {
      System.out.println("Caught Exception:  " + ioe);
    }
    catch (java.security.NoSuchAlgorithmException nsae) {
      System.out.println("Caught Exception:  " + nsae);
    }


    return true;
  }
}
```

Note
If SmartDongle access returns error codes 41 or 42, the problem may be that the mode of the dynamic SmartDongle device files in /proc/bus/usb/ and /dev/bus/usb/ do not allow user access.
As of linux kernel version 2.6.x, permissions can be set with hal facilities.
Refer to src/HAL/README.txt

Building the Examples

You must have Java Platform Standard Edition Development Kit and Java Standard Edtion Runtime Environment installed.

Download and unzip: SmartDongle JNI Examples.zip from the downloads section of www.smartdongle.com

Edit the example code to use your unique key values, for example:
private final static long p1 = 0xEC6CC589AEFD1E75L;
private final static long p2 = 0xFCEC0A6A82747B3FL;

Edit build.cmd file in the corresponding example's directory:

This variable corresponds to your main class name
MAIN_CLASSNAME=Example1

This variable corresponds to your full qualified class name
FULLY_QUALIFIED_CLASSNAME=Java_Example1
The fully qualified class name is required for the JNI function names.
If this is an incorrect fully qualified classname, executing example1 will produce this error:
Exception in thread "main" java.lang.UnsatisfiedLinkError: smartDongleWrite

Edit these paths:
The absolute path to the project directory such as:
WORKING_DIR="/home/your_home_directory/SmartDongle JNI Examples"

Path Java Development kit binaries
JAVA_HOME=/opt/sun-jdk-1.5.0.13

Path to GNU gcc complier:
CC=/usr/bin/gcc

Running build.cmd script, this will automate these steps:

Initial compilation of the main class and creation of JNI C header file.
SmartDongle interface source files are processed and SmartDongle JNI library (libuskjni.so) is compiled using your fully qualified class name (FULLY_QUALIFIED_CLASSNAME).
Computation of SHA fingerprint of libuskjni.so
The SmartDongle code in the main class file is updated with the SHA fingerprint.
The main class is then recompiled.