Certificates and Private Key Permissions

The information in this article is intended for software developers.

- This code can be used to troubleshoot private key issues with certificates in the Windows certificate store. It should run in .NET 3.5 and above.
- It will test CNG and legacy keys. Note Windows Server 2008 and Windows Vista and later create requests for CNG keys by default (if you're not sure, it's probably a CNG key). You need to set the variables for the appropriate store, serial number, and which type of keys to test for.
- Don't try using the debugger to view the PrivateKey property on CNG keys. It will always fail. That's why we use the key to authenticate to test.
- This code makes a HTTP request to https://iam-ws.u.washington.edu:7443 The behavior is undefined if access to that URL isn't available.
- There is a small possibility that the following error on a CNG key: "Private key exists, but we don't have permission to read it." is caused by a different problem. But check the private key first.
- Private key permissions can be managed by right-clicking a cert in the certificate manager > All Tasks and then click "Manage Private Keys...". Windows User Access Control (UAC) prevents unprivileged users from gaining programmatic access to the private key, even if they are a member of the local administrators group. So generally the user that runs your application (you, IIS or SYSTEM or whatever) will need to be given explicit permissions on the key. Note on Windows 10 and Server 2012, the "Manage Private Keys..." option is not available for certificates in the Current User ("Personal") store.

```
using System;
using System.Security.Cryptography.X509Certificates;
using System.Web;
using System.Net;
using System.IO;
using System.Security.Cryptography;
namespace certtest
{
    class Program
    {
        static void Main(string[] args)
        {

            //Select certificate location--note this tools always uses the "Personal" cert store in whatever
            //location you choose

            //local machine store
            StoreLocation certstore = StoreLocation.LocalMachine;
            //current user store
            //certstore = StoreLocation.CurrentUser;

            //set true/false below for types of keys you want to test
            bool testLegacy = true; //false;
            bool testCNG = true; //false;

            //Enter serial numbers for legacy and/or CNG keys here
            //Don't enter spaces show in cert manager properties
            string legacyserial = "40e9";  //Legacy key
            string cngserial = "5ab1";  //CNG key

            if (testLegacy)
            {
                Console.WriteLine(TestLegacy(legacyserial, certstore));
            }
            if (testCNG)
            {
                Console.WriteLine(TestCNG(cngserial, certstore));
            }
        }

        static string TestLegacy(string serial, StoreLocation certstore)
        {
            X509Certificate cert = null;
            X509Store store = new X509Store(StoreName.My, certstore);
            store.Open(OpenFlags.ReadOnly);
            X509Certificate2Collection col = store.Certificates.Find(X509FindType.FindBySerialNumber,
                                           serial, true);
            if (col.Count == 0)
            {
                return "could not find legacy cert with serial number " + serial;
            }
            try
            {
                cert = col[0];
                X509Certificate2 samecert = (X509Certificate2)cert;
                if (samecert.HasPrivateKey)
                {
                    AsymmetricAlgorithm key = samecert.PrivateKey;
                }
                else
                {
                    return "Cert has no private key!";
                }
            }
            catch (CryptographicException)
            {
                return "Private key exists, but we don't have permission to read it.";
            }

            string currentUser = System.Security.Principal.WindowsIdentity.GetCurrent().Name;
            return "This cert has a private key and this user, " + currentUser + ", was able to access it.";

        }
        static string TestCNG(string serial, StoreLocation certstore)
        {
            /*
             Note this method tests CNG cert private keys by trying to open a remote connection authenticating
using the cert.
             .NET doesn't handle CNG keys very well and this is the easiest way to test it
             */
            X509Certificate cert = null;
            X509Store store = new X509Store(StoreName.My, certstore);
            store.Open(OpenFlags.ReadOnly);
            X509Certificate2Collection col = store.Certificates.Find(X509FindType.FindBySerialNumber,
                                       serial, true);
              if (col.Count == 0)
            {
                return "could not find legacy cert with serial number " + serial;
            }
            try
            {
                cert = col[0];
                //Don't use the debugger to try to inspect the PrivateKey property on the samecert object.
                //It will throw an exception even if you have permission.  .NET limitation.
                X509Certificate2 samecert = (X509Certificate2)cert;
                if (samecert.HasPrivateKey)
                {
                    //rsaobj = samecert.GetRSAPrivateKey();
                    string uri = "https://iam-ws.u.washington.edu:7443/group_sws/v2/group/u_nebula_gca";
                    HttpWebRequest _request = (HttpWebRequest)HttpWebRequest.Create(uri);
                    _request.AllowAutoRedirect = true;
                    _request.Method = "GET";
                    _request.ClientCertificates.Add(cert);
                    WebResponse webResponse = null;
                    HttpWebResponse _response = null;
                    try
                    {
                        webResponse = _request.GetResponse();
                        _response = (HttpWebResponse)webResponse;
                    }
                    catch (WebException ex)
                    {
                        if (ex.Message == "The request was aborted: Could not create SSL/TLS secure channel.")
                        {
                            return "Private key exists, but we don't have permission to read it.";
                        }
                        else
                        {
                            return "something went wrong with this SSL client authentication attempt, but " +
                                "we're not sure what.  The error from this code was " + ex.Message +
                                 " and the error (if any) from the remote server was " + ex.Response;
                        }
                    }
                }
                else
                {
                    return "Cert has no private key!";
                }
            }
            catch (CryptographicException ex)
            {
                return "Unknown crypto error: " + ex.Message;
            }

            string currentUser = System.Security.Principal.WindowsIdentity.GetCurrent().Name;
            return "This cert has a private key and this user, " + currentUser + ", was able to access it.";

        }
    }
}
```