

## SSL/TLS for IntraWeb

by Henrick Hellström,

Copyright © 2004 StreamSec, All Rights Reserved

This document will help you getting started using StreamSec Tools for SSL/TLS with IntraWeb StandAlone Applications.

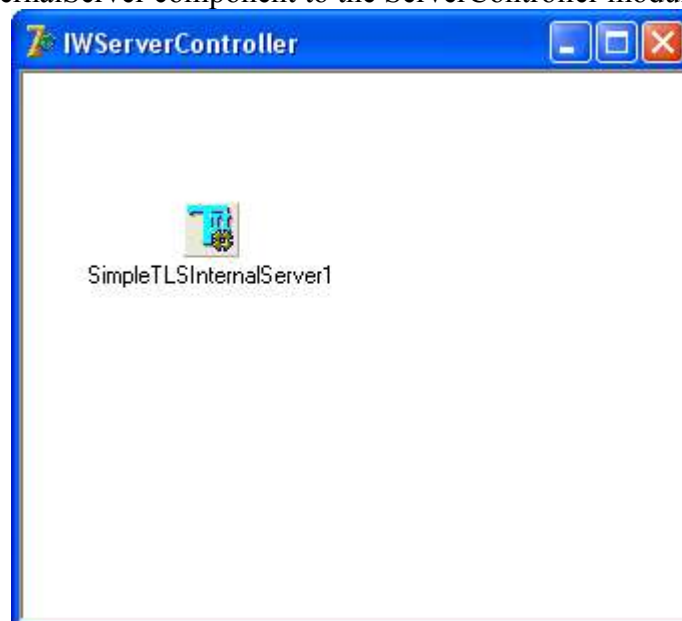


Open the ServerController unit. For IntraWeb 6 and IntraWeb 7, add SsTLSIn60IOHandler to the uses clause:

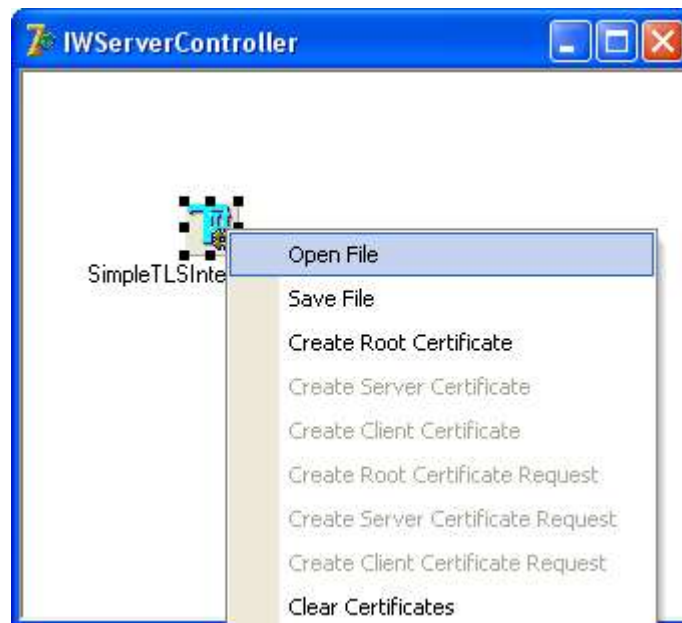
### uses

```
SysUtils, Classes, IWServerControllerBase, IWBaseForm, HTTPApp,  
SsTLSIn60IOHandler,  
// For OnNewSession Event  
UserSessionUnit, IWApplication, IWAppForm;
```

Add a TSimpleTLSInternalServer component to the ServerController module:



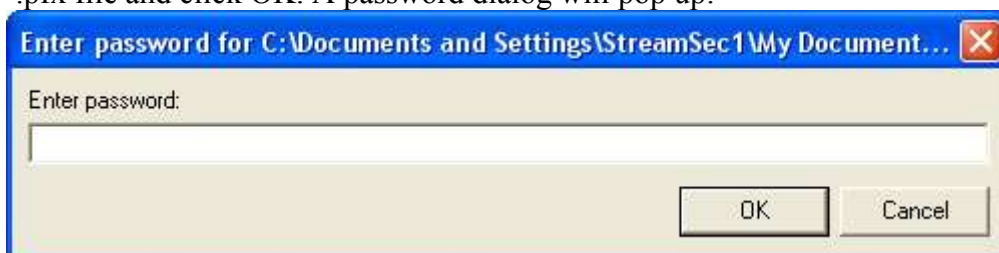
Right click on the SimpleTLSInternalServer component. Select Open File:



Open the \*.pfx file containing a Server Certificate created with StreamSec CertMgr:



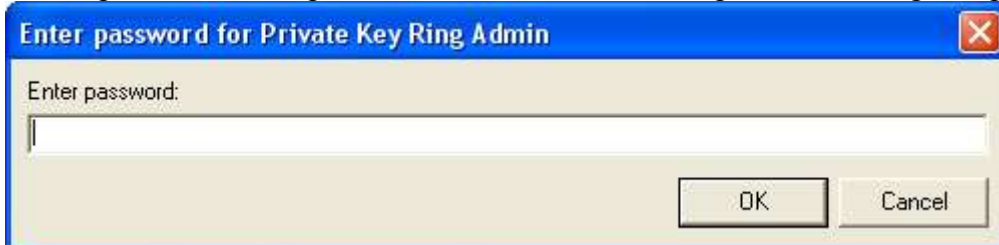
Select the \*.pfx file and click OK. A password dialog will pop up:



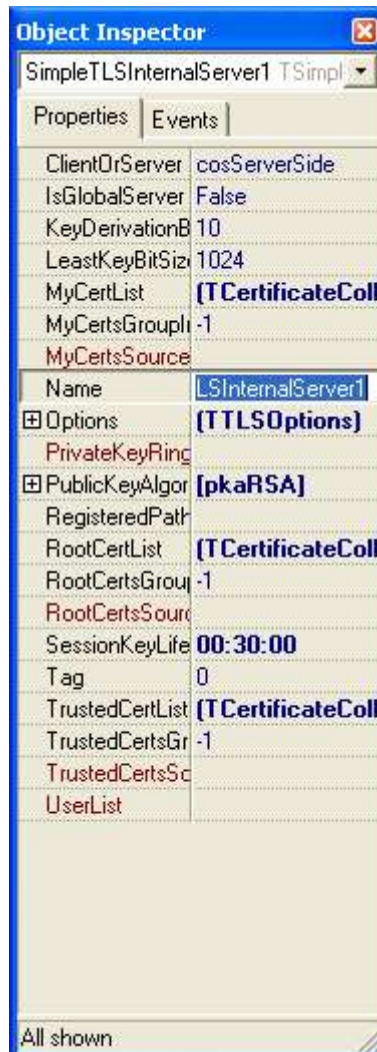
Enter the same password as when you saved the \*.pfx file. Click OK. A new password dialog will pop up:



This password will be used for protecting the \*.dfm resource containing the private key you just loaded from the \*.pfx file. Enter a password and click OK. A new password dialog will pop up:



Each private key ring is protected by two passwords. This password is used for preventing write access to the private key ring. Enter a password and click OK. The contents of the file will now be loaded into the SimpleTLSInternalServer component. As a result the PublicKeyAlgorithms property of the component will change to reflect this:



The value of the PublicKeyAlgorithms property indicates that the loaded certificate and private key can be used for server authentication using the RSA algorithm.

Save the project. (If you are using StreamSec Tools 2.1.8.190 or earlier you MUST save the project

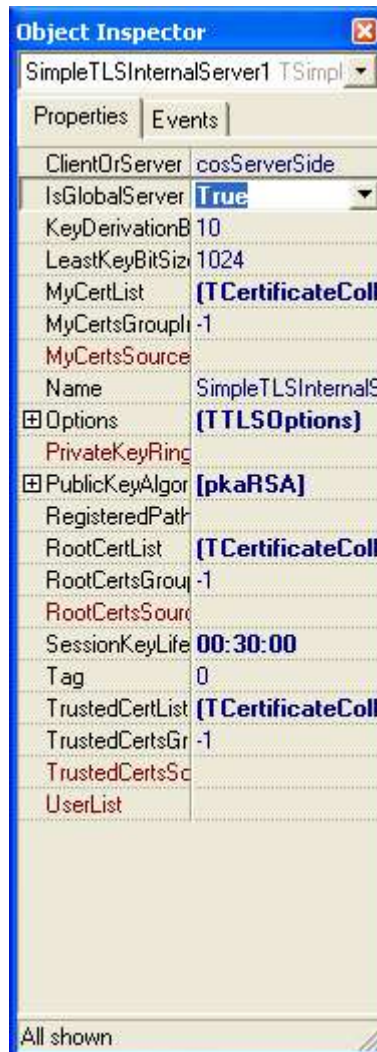
at this point, in order to circumvent a minor bug in the component streaming.)

A trick that will help you avoid having a password dialog pop up each time you run the application is to use the OnPassword event. Add SecUtils to the uses clause of the ServerController interface section. The OnPassword event handler might look like this:

```
procedure TIWServerController.SimpleTLSInternalServer1Password(  
    Sender: TObject; Password: ISecretKey);  
begin  
    Password.SetKeyStr('abc');  
end;
```

NOTE: While this is *common* practice it is not necessarily *good* practice, except during testing. By hard coding the password this way an adversary will only have to get access to the exe file of the server application in order to obtain the plain text private key of the server certificate.

The next step is to enable the component to be used for IntraWeb SSL/TLS. Change the value of the IsGlobalServer property to True:



There can be only one TLSInternalServer, SimpleTLSInternalServer or X509CertificateAuthority component per project with this property set.

Next, expand the Options property and disable RequestClientCertificates and RequireClientCertificates:



If you do not disable these properties the client user will get an error message telling him or her that the server requires client authentication but that the browser was unable to find a client certificate that could be used for this purpose.

If you want to use SSL/TLS for client authentication you must do either of two things:

1. Issue a client certificate to each user you want to give permission to enter your site.
2. Add the certificate of any CA you trust for issuing client certificates for your user. It is recommended that you firstly import all of these CA certificates with CertMgr and save them to a single \*.scl file you open with the SimpleTLSInternalServer component.

Lastly, select the ServerController module and change the SSLOptions.NonSSLRequest property to nsBlock. (Confer the IntraWeb documentation and news groups for more information about how to force some but not all pages of an IntraWeb application to be secure.)

Run the stand alone application, click the button for SSL and launch your default browser. The following message will pop up:



Go figure. This will happen each time you use a certificate you have issued with an inhouse CA unless you add the CA certificate to the certificate store of your browser. **IMPORTANT: Do NOT** over use your ability to add certificates to the certificate store of your browser. Even if it is your own certificate there is a risk that the certificate later gets compromised; in particular if it is a test certificate you don't treat carefully.