Embed Analyzer in a Hosting HTML Page
For
Single-Sign-On (SSO)

This document describes the steps to embed an IFrame that contains the Analyzer website in the hosting HTML page in a Single-Sign-On (SSO) implementation.

The concept is that user would sign-on through the hosting authentication page (SSO page) then access Analyzer directly on that page. After user has logged in to the hosting HTML page, the hosting page will generate an Analyzer URL along with a dynamic password to be stored in the application state. When Analyzer URL is executed along with the user name and dynamic password, Analyzer then calls web service function Login() to have the host re-authenticate the dynamic password stored in the application state to ensure the state is still valid. Once authenticated the Analyzer is then launched.

1. Setup IFrame
   First create an IFrame inside the SSO/Authentication page, this is for use to embed Analyzer.

   ```
   <iframe name="TopWindowOfAnalyzer" src="<%= analyzerUrl %>" > </iframe>
   ```

   name is the name of the IFrame, "TopWindowOfAnalyzer" is a static name that Analyzer used to identify its own frame/IFrame.

   src is the URL for the target Analyzer site, this is generated dynamically on the server side (see below).

2. Generate Analyzer URL
   At the IFrame webpage described above add the following code to generate the Analyzer URL

   ```
   protected string analyzerUrl = string.Format("{0}LoginAction.aspx?userid={1}&password={2}",
                AnalyzerUrlRoot,
                Session["UserId"],
                TempTicketTool.RequestDynamicPassword(false)
       );
   ```

   The ***password*** should *not* be user's real password, this is dynamically generated by TempTicketTool RequestDynamicPassword().

   Copy class TempTicketTool code listed in Section A1 below to SSO Web project.

3. Generate Password
   Once the user is authenticated by the SSO logic, SSO should create a ticket or a dynamically generated password to indicate the current user has been authenticated. Dynamically generated password should start with the prefix of "dnmcpwd_" then follow by the logon time. By using a static prefix of "dnmcpwd_" to indicate this is a dynamically generated password so it can be processed differently from regular password. The dynamic generated password will be stored in Application State to indicate that user has been authenticated and logged-in.

Please refer to Section A1 class TempTicketTool RequestDynamicPassword() method for sample to generate the dynamic password.

4. Login Analyzer
   Analyzer will bypass login screen and use *userid* and *password* supplied by the URL if they are included. To authenticate the userid and password Analyzer calls web service function Login() implemented by host SSO to validate the user credential. This re-authentication is needed since anyone could have provided an Analyzer URL with user name and password so Analyzer has to make sure this is a user currently authenticated by the host SSO program.

   The login() function needs to add the following code to support authentication of the dynamic password

```
if (password.StartsWith("dnmcpwd_"))
        {
           isValid = TempTicketTool.ValidateDynamicPassword(userId, password);
        }
```

   Dynamic password with the prefix of "dnmcpwd_" will be handled by ValidateDynamicPassword() in TempTicketTool, once authenticated the user can now enter Analyzer. For complete Login() function please see Section A2 below class ExtAuthServer Login() method.

5. Authenticate Dynamic Password
   Since dynamic password is generated from the SSO (host) system and stored in the application state so to authenticate simply check application state to verify whether or not the user has been authenticated. Please refer to class TempTicketTool ValidateDynamicPassword() method in Section A1 below for code example.

6. Remove Dynamic Password
   After user logout of SSO/host app or after session expiration the dynamic password should be removed from Application State. Add the following code to Session_End event in Global.asax.cs. Please see Section A3 class Global for complete Session_End event.

```
string key = "dnmcpwd_" + Session["UserId"];
        Application.Remove(key);
```

A1

```
   public class TempTicketTool
   {
      internal static string RequestDynamicPassword(bool renew)
      {
         string userId = (string)HttpContext.Current.Session["UserId"];

         string key = "dnmcpwd_" + userId;
         HttpApplicationState app = HttpContext.Current.Application;

         if (renew || app[key] == null)
         {
            string password = "dnmcpwd_" +
DateTime.Now.ToString("sHms").GetHashCode().ToString("x");
            app[key] = password;
            }

            return app[key].ToString();
        }

        internal static bool ValidateDynamicPassword(string userId, string password)
        {
            string key = "dnmcpwd_" + userId;
            HttpApplicationState app = HttpContext.Current.Application;

                return app[key] != null && app[key].ToString() == password;
        }
    }
```

A2

```
    public class ExtAuthServer : System.Web.Services.WebService
    {
        [WebMethod]
        public int Login(string userId, string password)
        {
                try
                {
               bool isValid = false;

               if (password.StartsWith("dnmcpwd_"))
               {
                   isValid = TempTicketTool.ValidateDynamicPassword(userId, password);
               }

               else
               {
                   isValid = ValidateUser(userId, password);
               }
                       return isValid ? 0 : -1;
                }
                catch
                {
                       return -1;
                }
        }

             // other web methods

    }
```

A3

```
    public class Global : System.Web.HttpApplication
    {
        protected void Session_End(object sender, EventArgs e)
        {
            string key = "dnmcpwd_" + Session["UserId"];
            Application.Remove(key);
        }
             // other event handlers
```

```
}
```