Using Apache mod_ntlm for SSO in Oracle Application Express (Proof-of-Concept)

Table of contents

1 Introduction

mod_ntlm is a module that is available for the Apache web server. It is available for both Windows and Unix versions. To use with the Oracle HTTP Server we will need the build for Apache version 1.3.

1.1 Links for additional information

- Oracle Apex Discussion Forum: http://forums.oracle.com/forums/forum.jspa?forumID=137

- Information about mod_ntlm can be found on:
  The sourceforge site http://sourceforge.net/projects/modntlm/

- Information about mod_ntlm with Oracle Apex can be found on:
  http://htmldb.oracle.com/pls/otn/f?p=18326 → Utilities →
  Using mod_ntlm with Oracle HTML DB

- Information at Twiki and Plone about installing mod_ntlm at Windows can be found on:
  http://twiki.org/cgi-bin/view/Codev/WindowsInstallModNTLM
  http://plone.org/documentation/how-to/singlesignonwindowsdomains

1.2 Further information

greenIT Est.
www.greenit.li | info@greenit.li

ORACLE PARTNER

2 Installation

2.1 Stop the HTTP Server (Apache Web server)

```
cd <Apex home>\opmn\bin
opmnctl stopall
```

2.2 Download mod_ntlm

nod_ntlm-1.3.zip from http://www.gknw.net/development/apache/apache-1.3/win32/modules/ for Apache 1.x

see also http://modntlm.sourceforge.net/

2.3 Extract mod_ntlm.so from the archive

You don't need any of the other files.

2.4 Put mod_ntlm.so into modules\

To location: <Apex home>\Apache\Apache\modules\

2.5 Edit http.conf file

At location: <Apex home>\Apache\Apache\conf\

At the end of the list of `LoadModule` commands, add the line:

```
LoadModule ntlm_module modules/mod_ntlm.so
```

At the end of the list of `AddModule` commands add the line:

```
AddModule mod_ntlm.c
```

Check for `KeepAlive On`

Locate the directive for MSIE and comment it out:

```
#SetEnvIf User-Agent ".*MSIE.*" \
#         nokeepalive ssl-unclean-shutdown \
#
```

2.6 Extend the Directive in dads.conf for Apex

File: <Apex home>\Apache\modplsql\conf\dads.conf
Extent the Location /pls/htmldb at the beginning (see also the directives file in mod_ntlm-1.3.zip):

```
AuthType NTLM
AuthName "NTLM authentication"
NTLMAuth On
NTLMAuthoritative On
NTLMOfferBasic On
NTLMDomain nb1
require valid-user
```

NTLMDomain: Set to the name of the domain you want users authenticated against for basic authentication.

2.7 Restart the HTTP Server

```
cd <Apex home>\opmn\bin
```

```
opmnctl startall
```

Look at the log files:

<Apex home>\opmn\logs\HTTP_Server~1

<Apex home>\Apache\Apache\logs\error_log.*

<Apex home>\Apache\Apache\logs\access_log.*

3 Test SSO with Apex

We will now need to test to see that our user details have been set correctly.

3.1 Create a Test Procedure in the Database

```
CREATE OR REPLACE PROCEDURE helloworld
IS
BEGIN
  htp.p('<html><head></head><body>');
  htp.p(owa_util.get_cgi_env('REMOTE_USER'));
  htp.p('</body></html>');
END helloworld;
/

GRANT EXECUTE ON HELLOWORLD TO PUBLIC;
```

3.2 Test with the Browser:

http://nb1:7777/pls/htmldb/martin.helloworld

You should see displayed the username you authenticated with at Windows.

Note:

http://nb1:7777/pls/htmldb/ - Apex Login Page

martin.helloworld – schema.procedure in the Database

3.3 Test NTLM Authentication in an Apex Application

3.3.1 Create an Authentication Function in the Database

```
CREATE OR REPLACE FUNCTION modNtlmPageSentry(pApexUser IN VARCHAR2 DEFAULT
'HTMLDB_PUBLIC_USER')
RETURN BOOLEAN
IS
  vAuthenticatedUsername  VARCHAR2(512);
  vCurrentSessionId       NUMBER;
BEGIN
  -- Get Authenticated User.
  vAuthenticatedUsername := UPPER(owa_util.get_cgi_env('REMOTE_USER'));
  -- Check to ensure that we are running as the correct database user.
  IF USER ^= UPPER(pApexUser) THEN
    RETURN FALSE;
  END IF;
  IF vAuthenticatedUsername IS NULL THEN
    RETURN FALSE;
  END IF;
  -- Get SessionId.
  vCurrentSessionId := wwv_flow_custom_auth_std.get_session_id_from_cookie;
  -- Check Application Session Cookie.
  IF wwv_flow_custom_auth_std.is_session_valid THEN
    htmldb_application.g_instance := vCurrentSessionId;
    -- Check Authenticated User --> Username from wwv_flow_session$ for
    --   current Session.
    IF vAuthenticatedUsername = wwv_flow_custom_auth_std.get_username THEN
      wwv_flow_custom_auth.define_user_session(p_user => vAuthenticatedUsername,
        p_session_id => vCurrentSessionId);
      RETURN TRUE;
    ELSE
      -- Unset the Session Cookie and redirect back here to take other branch.
      wwv_flow_custom_auth_std.logout(p_this_flow => v('FLOW_ID'),
        p_next_flow_page_sess => v('FLOW_ID') || ':' || NVL(v('FLOW_PAGE_ID'), 0)
        || ':' || vCurrentSessionId);
      -- Tell Apex Engine to quit.
```

```
      htmldb_application.g_unrecoverable_error := TRUE;
      RETURN FALSE;
    END IF;
  ELSE
    -- Application Session Cookie not valid --> Define a new Apex Session.
    wwv_flow_custom_auth.define_user_session(p_user => vAuthenticatedUsername,
      p_session_id => wwv_flow_custom_auth.get_next_session_id);
    -- Tell Apex Engine to quit.
    htmldb_application.g_unrecoverable_error := TRUE;
    IF owa_util.get_cgi_env('REQUEST_METHOD') = 'GET'  THEN
      wwv_flow_custom_auth.remember_deep_link(p_url => 'f?' ||
        wwv_flow_utilities.url_decode2(owa_util.get_cgi_env('QUERY_STRING')));
    ELSE
      wwv_flow_custom_auth.remember_deep_link(p_url => 'f?p=' ||
        TO_CHAR(htmldb_application.g_flow_id) || ':' ||
        TO_CHAR(NVL(htmldb_application.g_flow_step_id, 0)) || ':' ||
        TO_CHAR(htmldb_application.g_instance));
    END IF;
    -- Register the Session in Apex Sessions Table, set Cookie, redirect back.
    wwv_flow_custom_auth_std.post_login(p_uname => vAuthenticatedUsername,
      p_session_id => nv('APP_SESSION'), p_flow_page => htmldb_application.g_flow_id
      || ':' || NVL(htmldb_application.g_flow_step_id, 0));
    RETURN FALSE;
  END IF;
END modNtlmPageSentry;
/
```

3.3.2 Make a Test Apex Application using NTLM Authentication

- Go to Shared Components → Authentication
- Create a new Authentication Scheme

  Name:

  `NTLM`

  Page Session Management → Page Sentry Function:

  `RETURN martin.modNtlmPageSentry`

  Login Processing → Authentication Function:

  `-BUILTIN-`

  Logout URL → Logout URL:

  `wwv_flow_custom_auth_std.logout?`
  `p_this_flow=&FLOW_ID&p_next_flow_page_sess=&FLOW_ID:PUBLIC_PAGE:&SESSION`

  Make this scheme current

3.3.3 Run the Application

The application should start without needing to login and your APP_USER should now be your authenticated username. Apex will uppercase the username, be aware of this.