Sample Application: Call a Web Service with Single Sign On from a Windows Client Using a SAP Logon Ticket

Applies to:
- SAP NetWeaver 2004 SP 15 (and higher) and SAP NetWeaver 2004s SP 6 (and higher)
- mySAP Business Suite
- Microsoft Active Directory 2000 and 2003
- Microsoft Internet Explorer
- Microsoft Visual Studio

Summary
It is presented a sample application showing how to call a Web service with Single Sign-On (SSO) from a Windows client. The (non .NET) application first uses Windows Integrated Authentication to log on to an SAP NetWeaver Portal and obtain an SAP Logon Ticket. Then it calls a Web service on an SAP NetWeaver ABAP system presenting the obtained SAP Logon Ticket.

Author: Oliver Schmidt
Company: SAP AG
Created on: 16 November 2007

Author Bio
Oliver Schmidt is responsible for a Global Ecosystem and Partner Group initiative that enables Software Solution Partners (SSP) to provide their own Enterprise Services. Prior to joining the Market Development Engineering (MDE) team in 2006, Oliver was a key architect of the VM Container, SAP’s technology for enhancing the reliability and scalability of Java server applications. He has more than 20 years experience in software engineering and held several management positions at telecommunications and online services companies. Oliver earned a Dipl.-Inform.(FH) from Karlsruhe University for Applied Science.
# Table of Contents

Introduction ................................................................................................................................................... 3  
gSOAP .......................................................................................................................................................... 3  
WinInet .......................................................................................................................................................... 3  
The Sample Application ................................................................................................................................ 4  
Building the Sample Application ................................................................................................................... 8  
Running the Sample Application ................................................................................................................... 8  
Related Content ............................................................................................................................................ 9  
Copyright ..................................................................................................................................................... 10
**Introduction**


Another whitepaper [Sample Application: SSO with a .NET-based Web Service Client using SAP Logon Tickets](#) showed how to develop a .NET based Web services client that uses Windows Integrated Authentication and SAP Logon Tickets for Single Sign-On to a SAP NetWeaver Web service.

As a supplement to those two whitepapers I’ll present here a (non .NET) Windows client using Win32 APIs. Please note that only aspects that are actually different from the .NET client will be covered in this whitepaper.

**gSOAP**

Microsoft doesn’t offer a Win32 SOAP toolkit. So I chose the popular [gSOAP](#) because it is free, fast, lean and optionally supports the Windows Internet API.

**WinInet**

The [Windows Internet](#) (WinInet) API encapsulates the whole HTTP handling necessary in a Web service client like proxy server settings (incl. configuration scripts), SSL (incl. certificate management) and cookies (incl. persistent cookies).

Beside that WinInet supports Windows Integrated Authentication. When a Web server requires authentication and offers Keberos as authentication scheme WinInet (when running on Windows 2000 or above) will transparently authenticate based on the Keberos Ticket Granting Ticket received when the user logged on to Windows.
The Sample Application

The following command line sample application first obtains a SAP Logon Ticket and than calls the Z_BAPI_FLIGHT_GETLIST Web service:

```c
#define WIN32_LEAN_AND_MEAN

#include "Z_USCOREBAPI_USCOREFLIGHT_USCOREGETLISTSoapBinding.nsmap"
#include "gsoapWinInet.h"

void SoapEnrichSSO2(struct soap *Soap, char *Url)
{
    // Get access to the WinInet session that is going to be used for the
    // SOAP call. This might very well be called a hack as it assumes that:
    // - the WinInet plug-in has the id "wininet-2.0"
    // - the WinInet plug-in local data starts with a WinInet session handle
    // - the WinInet plug-in initialized this session handle on registration
    // These assumptions are true for gSOAP 2.7. They can easily be verified
    // by checking out gsoapWinInet.cpp.
    HINTERNET Session = *(HINTERNET*)soap_lookup_plugin(Soap, "wininet-2.0");

    // Contact an SAP Web AS Java with SPNegoLoginModule which issues a SSO2
    // ticket that is placed in an HTTP session cookie called MYSAPSSO2. This
    // cookie will be subsequently presented by the WinInet library for the
    // SOAP call.
    printf("\nObtaining SSO2 ticket...\n");
    HINTERNET Request = InternetOpenUrlA(Session, Url, NULL, 0, 0, NULL);

    // Provide feedback if something went generally wrong.
    if (Request == NULL) {
        char ErrorText[256];
        DWORD ErrorCode = GetLastError();
        if (ErrorCode != ERROR_INTERNET_EXTENDED_ERROR) {
            FormatMessageA(FORMAT_MESSAGE_FROM_HMODULE | FORMAT_MESSAGE_MAX_WIDTH_MASK,
                           GetModuleHandleA("wininet"), ErrorCode, 0, ErrorText, sizeof(ErrorText), NULL);
        } else {
            DWORD ErrorTextSize = sizeof(ErrorText);
        }
    }
}
```
void main(int argc, char *argv[]) {
    // Check for the obligatory command line argument.
    if (argc != 2) {
        printf("Usage: Win32Client <URL>\n" "\t<URL>: From where to obtain the SAP Logon Ticket\n");
        exit(EXIT_FAILURE);
    }

    // Create a gSOAP runtime environment (see comment section
    // in gsoapWinInet.h for details on the gSOAP flags usage).
    struct soap Soap;
    soap_init2(&Soap, SOAP_IO_DEFAULT | SOAP_IO_KEEPALIVE,

    // Get the SAP Logon Ticket from a specified URL.
    long ErrorCode = 0;
    long ErrorTextSize = 0;
    InternetGetLastResponseInfoA(&ErrorCode, ErrorText, &ErrorTextSize);
    printf("Error: %d %s\n", ErrorCode, ErrorText);
    exit(EXIT_FAILURE);

    // Check out the HTTP status code.
    DWORD StatusCode;
    DWORD StatusCodeSize = sizeof(StatusCode);
    HttpQueryInfoA(Request, HTTP_QUERY_STATUS_CODE | HTTP_QUERY_FLAG_NUMBER,
        &StatusCode, &StatusCodeSize, NULL);

    // Provide feedback if it's not the desired one and only.
    if (StatusCode != HTTP_STATUS_OK) {
        char StatusText[256];
        DWORD StatusTextSize = sizeof(StatusText);
        HttpQueryInfoA(Request, HTTP_QUERY_STATUS_TEXT,
            StatusText, &StatusTextSize, NULL);
        printf("Status: %d %s\n", StatusCode, StatusText);
        exit(EXIT_FAILURE);
    }

    // Release resources.
    InternetCloseHandle(Request);
}

// Sample Application: Call a Web Service with Single Sign On from a Windows Client Using a SAP Logon Ticket

SAP DEVELOPER NETWORK | sdn.sap.com
BUSINESS PROCESS EXPERT COMMUNITY | bpx.sap.com
© 2006 SAP AG
SOAP_IO_DEFAULT | SOAP_IO_KEEPALIVE | SOAP_IO_FLUSH);

soap_register_plugin(&Soap, wininet_plugin);

// Enrich the runtime environment with an SSO2 ticket.
SoapEnrichSSO2(&Soap, argv[1]);

// Create a request and a response message.
_RPC__BAPI_USCOREFLIGHT_USCOREGETLIST Request;
_RPC__BAPI_USCOREFLIGHT_USCOREGETLISTResponse Response;

// Populate the request message with the query parameters.
Request.AIRLINE = "LH";

RFC__BAPISFLDST DestinationFrom;
DestinationFrom.AIRPORTID = "";
DestinationFrom.CITY = "";
DestinationFrom.COUNTR = "";
DestinationFrom.COUNTR_USCOREISO = "";
Request.DESTINATION_USCOREFROM = &DestinationFrom;

RFC__BAPISFLDST DestinationTo;
DestinationTo.AIRPORTID = "";
DestinationTo.CITY = "Frankfurt";
DestinationTo.COUNTR = "";
DestinationTo.COUNTR_USCOREISO = "";
Request.DESTINATION_USCORETO = &DestinationTo;

// A FLIGHT_LIST needs to be present in the request message
// in order to have it present in the response message too.
RFC__TableOfBAPISFLDAT FlightList;
Request.FLIGHT_USCORELIST = &FlightList;

// Limit the response message size to a reasonable value.
int MaxRows = 20;
Request.MAX_USCOREROWS = &MaxRows;

// Go for it!
printf("\nCalling Web service...\n");
if (soap_call___RFC__BAPI_USCOREFLIGHT_USCOREGETLIST(
    &Soap, NULL, NULL, &Request, &Response) != SOAP_OK) {
    soap_print_fault(&Soap, stdout);
}
exit(EXIT_FAILURE);

// Check out the response message content.
printf("\nFlight(s) found: %d\n", Response.FLIGHT_USCORELIST->
__sizeitem);
for (int i = 0; i < Response.FLIGHT_USCORELIST->__sizeitem; i++) {
    printf("\nAirline: %s\n", Response.FLIGHT_USCORELIST->item[i]-
>AIRLINE);
    printf("Flight: %s\n", Response.FLIGHT_USCORELIST->item[i]-
>CONNECTID);
    printf("From: %s\n", Response.FLIGHT_USCORELIST->item[i]-
>CITYFROM);
    printf("To: %s\n", Response.FLIGHT_USCORELIST->item[i]->CITYTO);
}
}
Building the Sample Application
First you need to generate the client proxy from the WSDL file. With gSOAP this is a two step process I performed with the following commands:

```bash
wsdl2h -j -s -t RFC_typemap.dat Z_BAPI_FLIGHT_GETLIST.wsdl
soapcpp2 -C -L -w -x Z_BAPI_FLIGHT_GETLIST.h
```

The file RFC_typemap.dat used above contains only the following single line:

```plaintext
RFC = "urn:sap-com:document:sap:rfc:functions"
```

After that you can build the application as usual. Just be sure to include the following source files into your project:

```plaintext
<your gsoap dir>/stdsoap2.cpp
<your gsoap dir>/mod_gsoap\gsoap_win\wininet\gsoapWinInet.cpp
<your proxy generation dir>/soapC.cpp
<your proxy generation dir>/soapClient.cpp
```

Running the Sample Application
You need to provide the URL to obtain the SAP Logon Ticket from on the command line like this:

```bash
win32client.exe http://sapportal.mycompany.com:50000/irj/portal
```

If everything works as expected the output looks like this:

```
Obtaining SSO2 ticket...

Calling Web service...

Flight(s) found: 2

Airline: Lufthansa
Flight: 0455
From: SAN FRANCISCO
To: FRANKFURT

Airline: Lufthansa
Flight: 3577
From: ROM
To: FRANKFURT
```
Related Content

- Single Sign-On of Windows-based Web Service Clients using SAP Logon Tickets
- Sample Application: SSO with a .NET-based Web Service Client using SAP Logon Tickets
- gSOAP
- Windows Internet
Copyright

© Copyright 2006 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, Informix, i5/OS, POWER, POWER5, OpenPower and PowerPC are trademarks or registered trademarks of IBM Corporation.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies (“SAP Group”) for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

These materials are provided “as is” without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall not be liable for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials.

SAP does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third party web pages nor provide any warranty whatsoever relating to third party web pages.

Any software coding and/or code lines/strings (“Code”) included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, except if such damages were caused by SAP intentionally or grossly negligent.