Single Sign-On Assistant: An Authentication Broker for Web Applications

Fei Zhu
School of Computer Science and Technology
Soochow University
Suzhou, China, 215006
zhufei@suda.edu.cn

Hongjun Diao*
School of Computer Science and Technology
Soochow University
Suzhou, China, 215006
hjdiao@suda.edu.cn

Corresponding author: Hongjun Diao, hjdiao@suda.edu.cn

***Abstract*—Web-based systems are now widely used in many fields. Users are usually required to conduct identity authentication separately when logging in different systems before getting service. For the sake of security, it is improper to use a global identifier and password among several systems. Many approaches are proposed to solve the problem, among which single sign-on (SSO) is most popular schema with which a user logs in once and gains access to all systems without having to log in again. We put up a single sign-on assistant, called SSOA, for web-based applications. SSOA is an authentication broker and is implemented as plug-in installed in client side. When a user visits a web-based system using explorer, SSOA distills HTTP POST data, HTTP header used for login, reference address and authorization URI, and then constructs HTTP POST compatible data used for validation using the data returned by authentication broker server. Once a user is validated by SSOA, he can use systems and resources registered in SSOA by means of cached credential list. Due to the cached credential list, SSOA avoid adding excessive additional overhead and response time. SSOA communicates with authentication server via web service by SSL, thus obtaining as much generality as possible. SSOA achieves uniform identity authentication among heterogeneous systems, and attains most generality, simplicity and scalability with least cost as well.**

Keywords-authentication; single sign on; security; SSL; plug in

I. INTRODUCTION

Web-based systems are now widely used. Enterprises, faculties, institutes and organizations usually have their own systems. These systems generally require authentication when providing service. As a result, users have to remember accounts and passwords of each system, which is very inconvenient. Some have provided approaches to alleviate the trouble, such as auto-completion supported by Microsoft, with which account and password inputted by a user for a specific URI can be automatically stored. When the URI is visited again, the password can be automatically filled by system as the account is typed or selected. Although auto-completion brings us convenience, it is not of security in the case of a computer with multiple users.

We are developing a web-based digital campus system which mainly consists of network classroom and network-digital file cabinet. The system integrates four existing web application systems: news management system, video on demand system, bulletin board system and the laboratory management system. One problem occurred is that each system has its own user authentication system and validating processing logic. Modifying authentication modules and integrating them into a centralized user identity authenticating one will cause the relate data inconsistency, thus making the system do not work well. In addition, some applications are probable to be combined into the system in future. So it is necessary to leave an open and flexible interface for penitential incoming systems.

We put forward a single sign-on assistant for web-based applications, called SSOA, aiming at 'logging in once, running everywhere', with which we would solve uniform identity authentication among heterogeneous systems attaining simplicity, scalability and relatively low cost.

II. OVERVIEW OF SSOA SYSTEM

Single sign-on (SSO) is a property of access control of multiple, related, but independent software systems. With this property a user logs in once and gains access to all systems without being prompted to log in again at each of them. As different applications and resources support different authentication mechanisms, single sign-on has to internally translate to and store different credentials compared to what is used for initial authentication.

There are various kinds of identity authorization systems for web applications. Many take a HTTP POST approach which submits a group of related information to authorization component for processing. No matter what information is required by login page, POST data constantly are HTTP header plus name/value pairs. When the name/value pair is received, a server deals with validation according to predefined processing logic.

Hence, we can save HTTP POST data when a user logs in a system. When the user visits the page again, his/her request will be intercepted by the system, and then compose HTTP POST data after necessary processing. The data, afterwards, are sent to authorization components. As a result, user name and password inputting can be omitted. The login procedure can be executed by explorer monitoring program rather than the user.

As shown in figure 1, when a user visits a web system using Internet explorer, the system URI is send to authentication broker server after it is captured by a monitoring plug-in installed in Internet explorer client. In step 3, authentication broker server distills HTTP POST data, HTTP header used for login, reference address and authorization URI. In step 4 and step 5, plug-in of monitoring IE in client establishes data returned by authentication broker server and constructs HTTP POST

978-0-7695-3923-2/10 $26.00 © 2010 IEEE
DOI 10.1109/WKDD.2010.94

compatible data which will be sent to authorization component for users to log in. After that, the data are sent to authorization component to conduct validation. In step 6, a validated page is sent to the user.

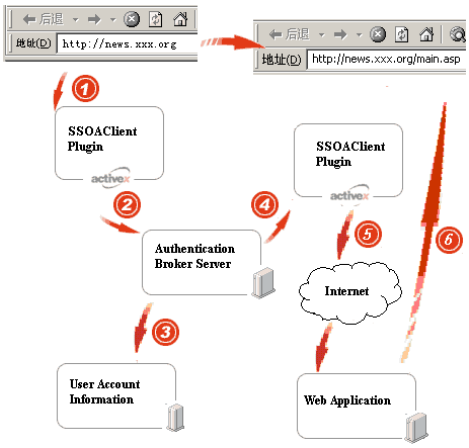

Figure 1. The overall view of login procedure using authentication broker.

III. DESIGN OF COMPONENTS IN SSOA

The system mainly consists of monitoring plug-in, authentication broker server, and user account information. In our system, the plug-in is implemented as a COM component which derives from interface IDispatch and implements interface IObjectWithSite. We register an item (HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Browser Helper Objects) in registration table so that SSOA can be automatically instanced each time Internet Explorer is started. Normally each IE process is related with a SSOA. As a result, SSOA can get all Internet Explorer operations and it can also add a hock for user-defined function to monitor Internet Explorer conducts.

SSOA intercepts the system URI, interacts with the authentication broker server and simulates login procedure. Authentication broker server is designed as a web service, which provides identity authorization, broker URIs adding and login data acquiring. We here show some key components of SSOA.

A. Component AccountInfo

Component AccountInfo is used to store basic and important information of user accounts. AccountInfo has five properties: uID, uName, Password, Role, and Online. uID is the unique identifier of user account, uName represents account name, Password stores password for user to log in, Role is used to represent user account role in the system and Online is used to show whether an account is online .

In AccountInfo, Password is stored in an encrypted style to ensure security. In the system, we use DES to encrypt the password. Therefore, when a user logs in, the password inputted should be encoded by DES before comparing with the stored password.

B. Component Credential

Authentication broker server will return an authentication credential to user after validation using account and password to log in the server. Component credential is used to store user credential information. It has four properties: crdID, uID, cIP and eTime. crdID is the credential identifier, uID represents account dentifier, cIP stands for current IP address of the machine user logged in, and eTime is used to figure out the expired time of user credential.

C. Component URIBroker

Component URIBroker is used to save user broker data. After validation, authentication broker server will get POST data, HTTP header, reference URI and authentication URI from URIBroker. Component URIBroker has six properties: uID, lgURI, pData, hData, rURI and aURI. Property uID stands for user's unique identifier, lgURI is the URI of the system for user to log in, pData is the HTTP post data, hData represents HTTP header data, rURI means reference URI and aURI is the URI for authentication.

As most of properties of URIBroker are critical to users, they should be encrypted by SSOA, e.g. using DES algorithm. In our system, the data are ciphered by DES using keywords as key.

IV. DESIGN OF AUTHENTICATION BROKER SERVER

SSOA is a plug-in which can be used in Microsoft Internet Explorer and Microsoft Windows browser, as well as those developed by other venders. Authentication broker service is a web service supplied by SSOA for users. SSOA client should log in authentication broker server to verify if the use can use the service. The processing logic is shown as follows.

Step 1. SSOA client connects SSOA server using Security Socket Layer(SSL);
Step 2. SSOA server gets account and password pair from SSOA client;
Step 3. SSOA server encrypts the account and password using DES algorithm;
Step 4. SSOA server compares the encoded data of account and password with the stored account and password pair;
Step 5. After passing validation, the server generates an authentication credential;
Step 6. SSOA server gets URI data from URIBroker in accordance with uID;
Step 7. SSOA server generates a credential list with the URI data attained which is a cache list used to speed up processing;
Step 8. A new item, mainly including credential, uID, cIP and eTime, will be inserted into component Credential.

Authentication credential of SSOA server is similar to session in web service. Once a user is validated, other systems registered in SSOA can use the credential to communicate with the server. Normally, a server will invalidate credential automatically if the user doesn't use it to access applications or resources registered in authentication broker server during a period of escaping time, e.g. 20 minutes. Authentication credential is shared by all through systems registered in SSOA, which is essentially different from the mechanism of session. The

credential is maintained by the authentication broker server. By the mechanism, the server can easily determine the role of a user.

Before inserting an item, SSOA will save POST data when accessing the URI and encrypt the data using DES algorithm. Afterwards, SSOA sends them to authentication broker server to add a new item. The processing logic is shown as follows.

Step 1. SSOA client connects SSOA server using Security Socket Layer(SSL);
Step 2. Server receives authentication credential and encrypted data concerning URI, pData, hData, rURI and aURI from SSOA client;
Step 3. Authentication broker server gets uID from component Credential according to crdID;
Step 4. Set eTime in component Credential as current time of server machine plus escaping time predetermined by the system;
Step 5. Authentication server inserts an inserting an item into component URIBroker, including uID, URI, pData, hData, rURI and aURI.

If the destination URI has been added into credential list, SSOA gets pData, hData, rURI and aURI data from broker authentication server.

Step 1. SSOA client connects SSOA server using Security Socket Layer(SSL);
Step 2. Server receives authentication credential and encrypted URI address from SSOA;
Step 3. According to authentication credential, authentication server gets uID from component Credential;
Step 4. Set eTime in credential as current time of server machine plus escaping time predetermined by the system;
Step 5. Get pData, hData, rURI and aURI data according to uID and encrypted URI address.

The server returns a credential list when SSOA client logs in authentication broker server. The credential list necessarily needs a mechanism to keep synchronization with the latest credential list stored in the SSOA server. The following is the processing logic that the SSOA requires to acquire broker URI data of users again.

Step 1. SSOA client connects SSOA server using Security Socket Layer(SSL);
Step 2. Server receives authentication credential from SSOA;
Step 3. Authentication server gets uID from component Credential using authentication credential;
Step 4. Data attained from component URIBroker are sent to SSOA;
Step 5. Set eTime in credential as current time of server machine plus escaping time predetermined by the system.

Authentication broker server also supports data management by users. User can apply to authentication server broker for a new account, modify password and manage existing broker URI.

Creating a new account and managing broker URI are relatively simple. Taking data security into account, all data are stored in a cyphered way, which, as a result, adds more trouble in password modification. The following is processing logic of password modification.

Step 1. The server receives account, old password, new password and confirmed new password from IE;
Step 2. The server conducts user validation using account and old password;
Step 3. After passing validation, the server judges whether the new password and confirmed new password are consistent;
Step 4. The server gets uID, URI, pData, hData, rURI data from component URIBroker according to uID;
Step 5. The server deciphers the data using old password as key;
Step 6. The deciphered data are then encrypted using new password as key;
Step 7. The server stores the newly encrypted data in component URIBroker.

V. CONCLUSION

As an integral part of information system, identity authentication is getting more and more important. Generally, users are required to input account identifier and password when logging in different web applications for service. For the sake of security, it is impossible to share a common identify and password among systems.

We put up an authentication broker for web-based applications, called SSOA. With SSOA, users do not have to repeat authenticating when entering in different web-based applications. SSOA gets user's POST data when he/she logs in some web application system the first time, and then constructs a credential by which SSOA simulates login procedure and conducts automatically login, thus eliminating user's manual identity authorization and therefore achieving single sign on.

We apply SSOA in a web-based digital campus system. With SSOA, we integrate four existing user authentication systems into a unified one in a loosely couple way without extensively modifying the authentication modules of the existing systems. By SSOA, we can log in once and use four heterogeneous systems conveniently. In addition, SSOA also provides interfaces for penitential incoming systems.

REFERENCES

[1] Fumiko Satoh, Takayuki Itoh, "Single Sign On Architecture with Dynamic Tokens," 2004 Symposium on Applications and the Internet (SAINT'04), 2004, pp.197.

[2] Cheng Yang, Jianbo Liu, Jiayin Tian, Yichun Zhang, "Authentication Scheme and Simplified CAS in Mobile Multimedia Broadcast," icmecg, 2009 International Conference on Management of e-Commerce and e-Government, 2009, pp.517-521.

[3] J.S. Park, R. Sandhu, "Binding identities and attributes using digitally signed certificates," acsac, 16th Annual Computer Security Applications Conference (ACSAC'00), 2000, pp.120.

[4] Jeremy Goold, Mark Clement, "Improving Routing Security Using a Decentralized Public Key Distribution Algorithm," icimp, Second International Conference on Internet Monitoring and Protection (ICIMP 2007), 2007, pp.8.

[5] Fei Zhu, Qiang Lv, "ACEAC: A Novel Access Control Model for Cooperative Editing with Workflow," isecs, 2008 International Symposium on Electronic Commerce and Security, 2008, pp.1010-1014.

[6] Enrique de la Hoz, Antonio García, Iván Marsá-Maestre, Miguel Ángel López-Carmona, Bernardo Alarcos, "An Infocard-Based Proposal for Unified Single Sign on," saint, 2009 Ninth Annual International Symposium on Applications and the Internet, 2009, pp.231-234.

[7] Fumiko Satoh, Takayuki Itoh, "Single Sign On Architecture with Dynamic Tokens," saint, 2004 Symposium on Applications and the Internet (SAINT'04), 2004, pp.197.

[8] Wei Huang, Jinhua Xu, "M4WebGIS: A Mobile Agent-Based Middleware for WebGIS", wcse, Second International Workshop on Computer Science and Engineering, 2009, pp.234-238.

[9] Fang Ying-lan, Han Bing, Li Ye-bai, "Research and Implementation of Key Technology Based on Internet Encryption and Authentication," 2009 International Conference on Networking and Digital Society, 2009, vol. 1, pp.179-182.

[10] Yan-zhi Hu, Da-wei Ma, Xiao-fei Li, "An Improved Authentication Protocol with Less Delay for UMTS Mobile Networks," 2009 International Conference on Networking and Digital Society, 2009, vol. 2, pp.111-115.

[11] Jeremy Goold, Mark Clement, "Improving Routing Security Using a Decentralized Public Key Distribution Algorithm," icimp, Second International Conference on Internet Monitoring and Protection (ICIMP 2007), 2007, pp.8.

[12] Jian Liu, Dianfu Ma, Zhuqing Li, Dou Sun, "A Formal Description of Web Services Container Architecture," 2009 Fourth International Conference on Internet and Web Applications and Services, 2009, pp.30-36.

[13] Jingjun Zhang, Fanxin Meng, Guangyuan Liu, "Research on SOA-Based Applications Based on AOP and Web Services," 2008 International Conference on Computer and Electrical Engineering, 2008, pp.753-757.

[14] Youxiang Duan, Yongtang Bao, Lijiang Pan, Beibei Yan, Jiuyun Xun, Nianyun Shi, "A Secure Web Services Model Based on the Combination of SOAP Registration Info and Token Broker," 2008 International Symposium on Computer Science and Computational Technology, 2008, vol. 2, pp.15-20.

[15] Jian Wu, Zhimin Huang, "Broker-Based Web Service Security," 2008 IEEE Asia-Pacific Services Computing Conference, 2008, pp.1282-1288.

[16] Udaya Kiran Tupakula, Vijay Varadharajan, Sunil Kumar Vuppala, "SBAC: Service Based Access Control," 2009 14th IEEE International Conference on Engineering of Complex Computer Systems, 2009, pp.202-209.

[17] Sebastian Rieger, "User-Centric Identity Management in Heterogeneous Federations," 2009 Fourth International Conference on Internet and Web Applications and Services, 2009,pp.527-532.

[18] Vipin Samar, "Single Sign-On Using Cookies for Web Applications," wetice, IEEE 8th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 1999, pp.158.