Research and Design of CAS Protocol Identity Authentication

HaoYuan Zhou
Communication university of China Security
big data processing and application
the key laboratory of Beijing
Beijing, China
18261187372
haoyuanzhou@yahoo.com

LIGU ZHU*
Communication university of China Security
big data processing and application
the key laboratory of Beijing
Beijing, China
18601137428
zhuligu@cuc.edu.cn

***Abstract*-With the rapid development of science and technology today, in the face of a variety of different systems, the use of a secure, unified and efficient authentication system can not only simplify the workload of administrators, but also simplify the memory of users and ensure the security of user data. We studied the advantages and disadvantages of several existing authentication methods. In order to facilitate multiple systems to achieve unified authentication login, we choose to use CAS to realize SSO's single sign-on function. It enables multiple subsystems to share a safe and reliable login method, and users can switch access freely among multiple subsystems. At the same time, in order to avoid the disadvantages caused by user's simple password, we improved the CAS encryption method with a small part of the custom encryption method. Some opinions and suggestions are put forward on the problem that single point of logout does not destroy session and users can log in after logout.**

Keywords: CAS; SSO; Security mechanism; Digital unification

I. INTRODUCTION

SSO is the abbreviation of "Single Sign On" in English. It separates the login logic of the user, and achieves the effect of entering the user name and password once to log in multiple products at the same time. SSO has obvious advantages in practical use. It can improve the user experience, avoid the repeated development of multiple systems and improve the safety factor of the system[1]. Therefore, it can be seen that SSO is not only useful but also necessary. For a further level of abstraction and partitioning, SSO is only responsible for logging into this single function, designed to meet a single responsibility principle[2]. SSO is just an architecture and a design, while CAS is a means to implement SSO. The relationship between the two is abstract and concrete. Of course, there are other ways to implement SSO besides CAS, such as simple cookies. Here we choose to use SSO. Similarly, there are many advantages to using SSO. It can improve the user experience, avoid repeated development when programmers write code, and improve the safety factor.

CAS officially became a project of ja-sig in December 2004.CAS is a central authorization service and is essentially an open source protocol. CAS was originally written by the Java language. According to different needs and programming languages, it can be applied on the application platform written by different languages to provide identity authentication services. CAS was originally designed to be a single sign-on authorization authority for different Web applications. It simplifies the complex process of user's identity authentication, reduces the mutual influence among the modules in the system, reduces the coupling degree among them, and enhances the security. CAS is also significant in terms of the advantages of use, and the configuration is simple, as you only need to configure it directly in the configuration file. Multiple authentication interfaces are supported. The coupling degree of CAS is very low, which reduces the coupling degree of authentication module in system design. Support for many languages such as Java, python, etc. The security performance is very high, using SSL protocol. SSL protocol provided by the service mainly includes, to achieve two-way authentication between the user and the server, to ensure the accuracy of data transmission. Data in transit is encrypted to prevent information leakage and theft. It can also be well supported in very complex network environment.

The architecture of CAS mainly includes two parts, the CAS server side and the CAS client side. The CAS server side requires a separate deployment process to implement centralized authentication for different users. Primarily responsible for login validation, redirection, and ticket validation for client requests. The CAS client is mainly responsible for authenticating the user's username/password, issuing tickets, etc. which needs to be deployed separately[3]. The CAS client is uniformly deployed with the Web application. When the user tries to log in the system to access the resources they need, the CAS client will send the request directly to the CAS server for request verification. The TGT(Ticket- Granting Ticket) is the login ticket issued by CAS for the user, and the user proves the successful login through the TGT. The CAS server generates a TGT and adds it to the user's session. The TGC(Ticket-Granting Cookie) is a unique identity and a credential that the CAS server uses to identify the user. ST is the unique service identification code when CAS logs in for users. When a user accesses the system, the system finds that the user does not have ST, then it will give feedback to the user to get ST from CAS[4]. After verification, users can be allowed to access the corresponding resources of the system.

978-1-7281-9481-3/20/$31.00 ©2020 IEEE
DOI 10.1109/CVIDL51233.2020.00-66

II. BACKGROUND

At present, there are many authentication technologies in abroad. The United States has vigorously developed FICAM digital identity authentication scheme based on KPI framework, which has been applied effectively in the field of government affairs. It is mainly for the United States to establish a unified identity, certificate and access management system in the classified system. In order to solve the cooperative work and identity authentication among the federal classified networks. However, the eu is actively promoting eID electronic identity authentication scheme, which is widely used in the civil field. One of the key tasks of the eu network development strategy is to establish a network trusted identity system and form a unified identity service market within the eu. FIDO Online Fast authentication standard is an open standard protocol proposed by Fast Identity Online Alliance to provide an Online authentication technology architecture with high security, cross-platform compatibility, particularly the good experience and strongly privacy of users. The MIT designers came up with the kerberos single sign-on protocol. When the web application requested by the user needs permission judgment, the interface of the service is invoked to judge the service and realize the safe and reliable network service[5]. IBM's Tivoli Access Manager[6]. Microsoft's.NET passport[7].

Admittedly, compared with foreign countries, China's IT technology is relatively backward, and there are no unified standards and rules for identity authentication. Currently, many identity authentication technologies in China use LDAP+PKI to achieve unified user authentication[8].

Public key infrastructure (PKI) is an infrastructure of network information security services provided by public key technology. PKI is implemented by managing secret keys and certificates to provide a relatively safe network environment for users and avoid attacks from hackers. This approach ensures the confidentiality and integrity of the data. PKI technology is commonly used in the field of information security. The composition of PKI is mainly composed of the following parts: CA is the most important part of PKI and is provided by reliable and authoritative third parties. Digital certificate library is used to store some important information which published by CA, and users can freely query to get the information they need. The secret key backup and recovery system is to prevent the user from losing the secret key, so that the password cannot be solved. Therefore, the system is provided to prevent the occurrence of accidents. Certificate invalidation system is mainly used when the user no longer continue to use or user identity changes, we need use the certificate invalidation system to process information. Application interface API is to provide an interface for a variety of applications, through this interface, a variety of different types of applications can carry out a safe and reliable way of information interaction, to provide convenience for the network security environment[9]. LDAP is essentially a very specialized database which is superior in terms of read and write performance. It is much faster to read than many relational databases. The LDAP server can manage all user information uniformly, accessing the same database each time.

III. MOTIVATION

With the rapid development of technology, both businesses and schools are highly dependent on computers to replace the tedious and cheap labor. In this period, various management systems emerge in an endless stream and show a rapid growth trend. For each system administrator, the database is used to store the user name and password. On the one hand, the system administrator needs to spend a lot of time to ensure the security of users and passwords. On the other hand, different administrative systems require different passwords for users. Especially, when the password is complicated, users are more likely to forget their password. When the password is relatively simple, it is easy to cause the user's information to leak. Therefore, in order to facilitate the management of users and passwords, it is necessary to establish an efficient and reliable security authentication system. After using the security authentication system, no matter the user logs on to the management system of several different applications, they only need to do one security authentication login, and they can switch among different system administration applications at will[10].

After studying several existing authentication technologies, we chose to use single sign-on(SSO) to ensure that only one master system needs to be logged in once and the rest of the subsystems can be accessed freely. In many approaches of single sign-on, the most popular is the CAS framework. CAS is an open source framework led by Yale university[11], so in this paper, we use the CAS framework for unified identity authentication. At the same time, CAS model has some security risks when used. In order to enhance its security, we improved the way of saving user passwords in the CAS model. CAS has two encryption methods, MD5 and SHA. In order to upgrade its security, we can customize the encryption algorithm. After we encrypted the user's password with MD5, we spliced the salt value what we set in the database and used MD5 to encrypt it again. In this way, the security of the user's password is upgraded. At the same time, we found in the experiment that there is a certain vulnerability in the single point logout of CAS. When a request is made to destroy the session, the session still can be logged in without being destroyed. Some solutions and ideas are proposed to solve this problem.

IV. SYSTEMDESIGN

The emergence of single sign-on technology adds new vitality to the mobile office system and opens up a new direction for the development of the future mobile office system[12]. The configuration of CAS is relatively simple, it can support the development of multiple languages. CAS is non-blocking, it can avoid deadlocks. The impact between threads is relatively small, there is no system overhead caused by lock competition, and there is no overhead of frequent scheduling between threads. Therefore, it is very convenient to use CAS technology to realize user identity authentication. The authentication process of CAS is Fig.1.

When the user accesses application system “a” for the first time, the filters on the CAS client receive web requests from the user. However, if the system find the user is not logged in, the

CAS client will redirect a request to the CAS server. At the same time, the redirected URL will use the service parameter to pass the user's destination address to a login interface of the CAS server. The CAS server will send a login interface that we have defined for the user to log in. After the user enters the user name and password on the login page, the user will send it back to the CAS server. Then the CAS service authentication is passed, the user will carry a TGC(one of the tickets) and the cookie value will be saved. The CAS server will generate a random Service Ticket based on the parameter service, which will be returned after adding the service parameter of the url, like the following url:

http://sso.aaa.cn/login?service=https%3A%2F%2Fwww.a.cn

The “AuthenticationFilter”on the client chooses to skip after receiving the Service Ticket and is processed by the “TicketValidationFilter”to access the “servicevalidate” interface on the CAS server side. The interface will verify the valid information of the “Service Ticket”. If the validation is successful, then the user can access the system for operation. Otherwise, it will force the user to log in before the operation.

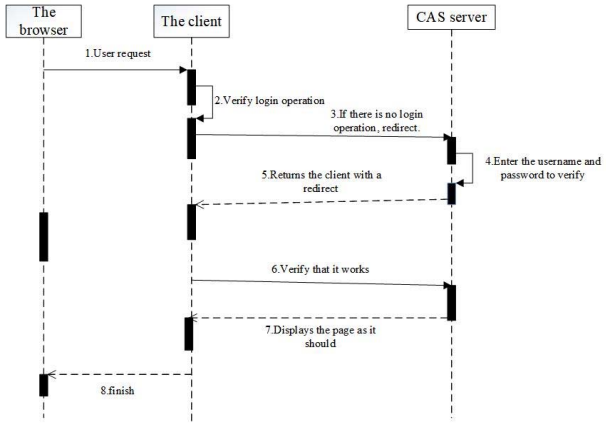

Fig. 1. The workflow of CAS

In the current network environment, the use of unencrypted password storage is vulnerable to cracking, so now in the development project will pay attention to the user password encryption. CAS chooses to use MD5 and SHA encryption algorithm in the part of handling user encryption. The MD5 algorithm is one of the Hash algorithms. MD5 is generally used in password encryption to protect the user's password security. The password stored in the database is not in clear text, but in an md5-encrypted manner. In this way, even if the user's information is stolen by hackers, hackers still can't get the original password of the user, for the user a layer of security. In the process of using, users are likely to set the password very simply in order to remember the password easily and quickly, such as "123456", "abcdefg", "6666666". In order to prevent the user's password from being easily cracked, we choose to use the password after salt value processing for secondary encryption, to ensure the security of the password. Here, we use the following encryption method: after the password entered by the user, it is encrypted by MD5 and a salt value in the database is spliced, and then MD5 is used for secondary encryption after splicing. To do what we need, we need to extend the encryption algorithm. First, we configure the encryption method of CAS, and select the user's encryption method "MD5PasswordEncoder" in CAS.

<propertyname="passwordEncoder" ref="MD5PasswordEncoder"></property>

CAS compares this encryption to the password retrieved from the database. We also need to configure the content related to MD5 encryption in the configuration file on the web side.

<beanid="MD5PasswordEncoder" class="org.jasig.cas.authentication.handler.DefaultPasswordEncoder">

So far, the simplest CAS that supports MD5 encryption is complete.

What we need is not only MD5 encryption, but also some custom operations. From the above analysis, we can see that the main configuration of the encryption algorithm is in the configuration file of the web. To extend the encryption, we need to customize an encryption class and simply assemble it. We create a new interface class that implements the Password Encoder of CAS, the salt is a salt value defined in our database, and use "EncryptUtil.getMD5Str(EncryptUtil.getMD5Str(password) + salt)", which is the most important part we need to encrypt the password twice. Here the custom encryption algorithm is implemented.

The functions in the single sign-out and single sign-on sections of CAS are comparable and similar. The flow chart of single point logout is as follows:

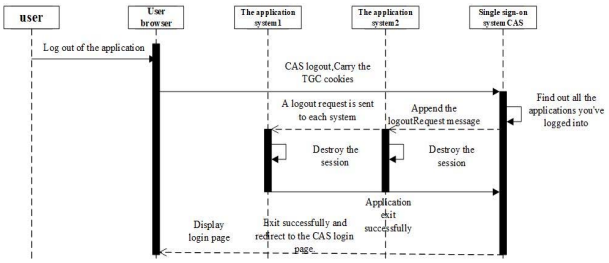

Fig. 2. The workflow of CAS logout

Make all CAS clients log out through the logout of the CAS server. The CAS server side logs out by sending a request "/logout". When the logout operation of the CAS service is requested, the CAS service removes the TGC carried by the client to find out all the applications which logged in. Send a “logoutRequest” to each application and append the “logoutRequest” message to destroy the session. After the application exits successfully, redirect to the CAS login page.

The flaw of CAS in the single-point exit is that the CAS server will issue the session request to destroy all the clients that have been registered in the CAS server. However, we found that the session was not destroyed in our actual use, and the single

point request did not take effect. In the case of fewer clients, we can first issue a request to the client, and then the client manually destroys the session before issuing an exit request to the CAS server. This solution is only suitable for the case of fewer clients. It is not applicable in the case of a large number of clients, so in the case of studying its source code according to the data, we found that the fundamental reason lies in the writing of the code. After getting the client addresses of all the logins, it determines whether the client addresses of those logins are valid. If the address registered at this time is localhost, it will be skipped directly, and many times, especially during development, we often test with localhost, which leads to errors. Therefore, in the development phase, when configuring CAS we do not write localhost but use its IP address 127.0.0.1 instead.

V. EVALUATION

In order to provide more convenient channels and ways for users to log in different systems, we use CAS to conduct unified authentication login management for users. No matter how many different subsystems we have, we don't need to set up a separate database for each subsystem to record its username and password. Our research work mainly includes the following aspects: using the identity authentication technology based on CAS protocol, introducing the architecture of CAS in detail, and deeply analyzing the CAS protocol and its working principle. At the same time, in order to better protect the user's password, the salt value is used to encrypt the password twice. It also addresses the problem of dealing with session not being completely deleted when CAS exits and enable users to better use CAS login master or sub system.

However, in fact, CAS protocol itself also has some shortcomings, and some security problems need to be solved urgently. The CAS authentication mechanism has been able to resist some potential security risks. However, in the case of a denial of service attack, CAS fails to respond effectively, the service is terminated, and some functionality is lost. This damage is very unfriendly to CAS. In addition, CAS also carries the risk of man-in-the-middle attacks. When the user requests to log in, authentication requirements are sent to the CAS server. After the CAS server completes the validation, it returns a request with an important ticket to access the service, according to the jump page we defined. In the process of redirection, HTTP protocol is adopted to carry out data transmission in the form of clear text, so it is easy to suffer from man-in-the-middle attack in the process of data transmission. Each time a user makes a request to access a page, the CAS server validates the information sent by the user. However, the user has no way to verify the authenticity of the CAS server, and only realizes one-way authentication. Although CAS can meet our needs of single sign-on, there are still some problems waiting to be solved.

ACKNOWLEDGMENTS

Supported by the Fundamental Research Funds for the Central Universities

REFERENCES

[1] Hua-Yan, P. , & Huan-Min, W. . (2014). Research and implementation of single sign-on platform based on cas. Computer Knowledge and Technology.

[2] Yu, J. , Wang, G. , & Mu, Y. . (2012). Provably Secure Single Sign-on Scheme in Distributed Systems and Networks. IEEE International Conference on Trust, Security & Privacy in Computing & Communications. IEEE.

[3] Wang, Y. , Wen, Q. , & Zhang, H. . (2010). A Single Sign-On Scheme for Cross Domain Web Applications Using Identity-Based Cryptography. Second International Conference on Networks Security Wireless Communications & Trusted Computing. IEEE.

[4] Yu Chang China. (2015). The students growth record system based on digital campus design and implementation. (Doctoral dissertation).

[5] Schmidt, & Leetta, M. . (2011). Illiad, cas, shibboleth, and php: the road to single sign-on. Journal of Interlibrary Loan Document Delivery & Electronic Reserve, 21(3), 149-156.

[6] Ye, Q. , Bai, G. , Wang, K. , & Dong, J. S. . (2015). Formal Analysis of a Single Sign-On Protocol Implementation for Android. International Conference on Engineering of Complex Computer Systems. IEEE Computer Society.

[7] Tao Y, Zhou F. The analysis and design for single sign-on in the mobile Application Data

[8] plus ota. (0). The single sign-on based on CAS and the design and implementation of a unified identity authentication system. (Doctoral dissertation).

[9] huang hua. (0). The unified identity authentication system based on Web application design and implementation. (Doctoral dissertation, university of electronic science and technology).

[10] Spoorthi, V. , & Sekaran, K. C. . (2014). Mobile single sign-on solution for enterprise cloud applications. International Conference on Networks & Soft Computing. IEEE.

[11] Spoorthi, V. , & Sekaran, K. C. . (2014). Mobile single sign-on solution for enterprise cloud applications. International Conference on Networks & Soft Computing. IEEE.

[12] Chen Y, Xia B, Wu B, et al. Design of web service single sign-on based on ticket and assertion[C]. International Conference on Artificial Intelligence, Management Science and Electronic Commerce. IEEE, 2011:297-300.