Open Journal of Applied Sciences, **2012, 2, 177-179**
doi:10.4236/ojapps.2012.23025 Published Online September 2012 (http://www.SciRP.org/journal/ojapps)

A Solution to Implement Dynamic Authentication

Bingbing Xia, Youping Dong
Information Technology Department, Shandong Jiao Tong University, Jinan, China
Email: jennifer_xiababy@yahoo.com.cn

Received July 6, 2012; revised August 5, 2012; accepted August 16, 2012

ABSTRACT

In view of the security risks of using static passwords to authenticate users, this paper gives a solution to implement two-factor authentication by using RSA token. A set of corresponding realization processes is proposed as well. Using dynamic password given by RSA can further verify user identity to improve the reliability of authentication.

Keywords: Token; Two-Factor authentication; NextToken

1. Introduction

Currently, most network users need to use user name and password to log on for authentication. Passwords used in systems, such as email system, online banking, are most static. But static passwords show more and more security risks with the development of hacking. Nowadays it is not a hard job either to obtain passwords by using Trojan or to decipher passwords by using hacking tools. Therefore, a more secure two-factor authentication mechanism is needed.

1.1. Two-Factor Authentication

Two-factor authentication is a cryptology concept, which is a strengthening authentication mechanism based on static password identification process. It uses physical tokens, such as dynamic password card, IC card and magcard, to further verify user identity so as to forbid unauthorized visitors, thereby to improve the reliability of authentication.

Three main components are included in a dynamic two-factor authentication solution that are a simple, easy-to-use token, a powerful management server and a kind of proxy software.

- Token: Different user has different token with which a random number is generated as the token code respectively. Usually the number changes every one minute so that it is valid only for a specific user at a particular time. As a dynamic password, token code further ensures the accuracy of user identification.
- Proxy software: When users log on, proxy software will send the request to the authentication engine of the management server. Logon can be successful provided that authentication process is verified.
- Management server: The management server uses the same algorithm and token code as the token to verify the token code.

Users can log on successfully when both password and token code are inputted correctly. Otherwise the token code will be requested once again. If three times of failure occurred, two following token codes need to be provided. Once the number of inputting exceeds a certain value, proxy software will lock the user account and prohibit it from logon again.

1.2. Introduction of RSA Token Authentication System

RSA token authentication system is a mature product and it generates a 6-digit or 8-digit number by applying the well known RSA algorithm [1]. The number changes every one minute and can never be repeated. Each user has a token that can generate the same 6-digit or 8-digit number by applying the same algorithm. Token code is inputted through client system and verification is carried through the server. User identity can be authenticated when the two numbers are the same. Otherwise the access will be denied. The advantages of this process are:

- The algorithm used is rigorous and hard to decipher.
- Synchronization between server and user token does not involve signal synchronization. Whereas systems that use SMS random code are restricted with signal strength limitations and may have delay.
- Special hardware is not needed for the client system. What users need to do is to input token code and no special interface, like USB or card reader, is required.

2. Solution

Though there have been some products that can embed RSA token product into operating system and VPN sys-

tem, corresponding development is not popular currently yet. Customers need to do extra interface development programming to realize two-factor authentication for usually this kind of product only provide a java API package [2]. Token as shown in **Figure 1** has a unique serial number for each user and the number is one-to-one corresponding to the user name in RSA system. By giving system user list to RSA server, one-to-one relationship between user name and token is built up [3]. The two systems can synchronize user identification through administrative processes but not underlying structure.

2.1. AuthUserBean.java Class

AuthUserBean.java class is created firstly in the interface development program based on the API given by providers. The variable userID represents user name and passCode represents token code. The status of logon is represented by the variable Status that has three values, value 0 means verification process is successful, value1 means it failed while value 2 means NextToken mode. Times of failure of inputting token code are calculated via function countPlus ().

The AuthUserBean.java class has two variables named as userID and passCode. Another variable is count used in the countPlus () method, and the method plus 1 to the count.

2.2. AuthManager.java Class

AuthManager.java class implements the communication with the RSA server with main function authUser (), in which the process of communication is created through the path parameter by using the API of RSA [4]. The code is as below:

```
AuthSessionFactory api = AuthSessionFactory.
getInstance (this. path);
AuthSession authSession = api.createUserSession ();
```

Next step is to determine the status of logon. If the status value is 2, it means nextToken mode that is a strategy of RSA server to prevent attacks [5]. When certain times of inputting failed, the server treats the logon as an attack and asks the user to input two consecutive token codes to change the status. If times of inputting wrong token codes continued reach a certain number, the user account will be locked. If the value of the Status is 0, the verification of user identity is successful; if the value of Status is 1, the times of wrong inputting needs to be recorded. The code is as below:

```
if (user.getStatus () == 2) {
status = user.getAuthSession ().next (user.getPassCode
());
user.setStatus (status);}
else if (user.getStatus () == 0) {
user.getAuthSession ().close ();}
else {
status = user.getAuthSession ().check
(user.getUserID (), user.getPassCode ());
user.setStatus (status);
if (user.getStatus () == 1) {user.countPlus ();}
if (user.getStatus () == 0) {user.getAuthSession ().close
();}}
```

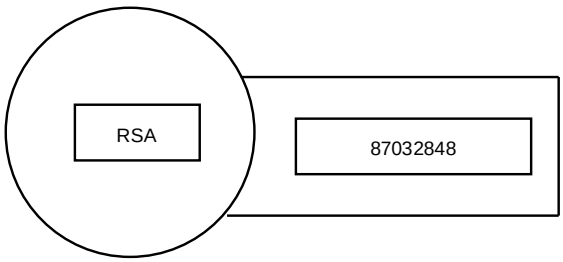

Figure 1. Token.

2.3. MainController Class

MainController class creates a servlet, the init () method is used to initialize the servlet，get the parameter path from the web.xml and judge the validity of the path. The main code of this method is as below:

```
String path = config.getInitParameter (“path”);
if (path == null) {
throw new ServletException (“path is null”);}
if (path. equals (““)) {
throw new ServletException (“path is blank”);}
```

The service () method is used to process the client’s request, action saves the input source identity, passCode saves the input RSA number. The main code is as below:

```
String action = request.getParameter (“action”);
String passCode = request.getParameter (“passCode”);
```

If the input source is incorrect, the program returns the login page:

```
RequestDispatcher dispatcher;
if (!”tokenUser”.equals (action)) {
dispatcher = request.getRequestDispatcher (“/login.jsp”);
dispatcher.forward (request, response) ;}
```

If the input source is correct, the program will turn to other verification,

- First, judge the existence or overdue of the session:

```
HttpSession session =request.getSession ();
if (session.getAttribute (“tokenuser”) == null) {
request.setAttribute (“logout”, “true”);
dispatcher = request.getRequestDispatcher
(“/login.jsp”) ;}
```

- Second, begin to verify the token number.

First, assign the token number named passCode to the user, then verify by the authUser () method. If the verification passed, the program will turn to result.jsp, else if the token number got is incorrect 3 times, the program will turn to the login page login.jsp.

```
AuthUserBean user = (AuthUserBean)
```



```
session.getAttribute (“tokenuser”);
user.setPassCode (passCode);
User = manager.authUser (user);
if (user.getStatus () ==0){
dispatcher = request.getRequestDispatcher (“/result.jsp”);
dispatcher.forward (request, response);}
else if (user.getStatus () ==1&&user.getCount () <3) {
dispatcher = request.getRequestDispatcher (“/login.jsp”);
dispatcher.forward (request, response) ;}
```

2.4. Web.xml File

The content that needs to be configured in web.xml is as below:

```
<Web-app>
<Servlet>
<servlet-name>MainController</servlet-name>
<servlet-class>servlet.MainController</servlet-class>
<Init-param>
<param-name>path</param-name>
<Param-value>
D:\\userlogin\\WEB-INF\\classes\\rsa_api.properties
</param-value>
</init-param>
</servlet>
<Servlet-mapping>
<servlet-name>MainController</servlet-name>
<url-pattern>/servlet.do</url-pattern>
</servlet-mapping>
<welcome-file-list>
<welcome-file>servlet.do</welcome-file>
</welcome-file-list>
</web-app>
```

2.5. Application Resources

The main part of the file rsa_api.properties is as follows:

- #RSA verification necessary file path created at the server port:
  SDCONF_TYPE=FILE
  SDCONF_LOC=D:\\userlogin\\sdconf.rec
  SDSTATUS_LOC=D:\\userlogin\\JAStatus.1
- #RSA verification file path after the first success dispath between the server and the WEB server:
  SDNDSCRT_TYPE=FILE
  SDNDSCRT_LOC=D:\\userlogin\\securid
- #WEB server log path:
  RSA_LOG_TO_CONSOLE=NO
  RSA_LOG_TO_FILE=YES
  RSA_LOG_FILE=D:\\userlogin\\rsa_api.log
  RSA_LOG_LEVEL=INFO

3. Conclusion

The process of two-factor authentication of user identity can be achieved through the dynamic password provided by RSA token. The reliability of verification is improved and all user information verified will be stored in the system for later use [6]. Through the process of token code validation, the accuracy of user identity is guaranteed thus to enhance the system security. It can be seen as a practical solution.

4. Acknowledgements

I would like to express my deepest gratitude to Tian Rui, who helped me a lot to complete this paper. Second, I will extend my heartfelt gratitude to teacher Dong You ping that helped me a lot during my work.

REFERENCES

[1] RSA Laboratories, “PKCS #15 v1.0: Cryptographic Token Information Format Standard [S]”.

[2] H. Krawczyk, M. Bellare and R. Canetti, “HMAC: Keyed-Hasing for Message Authentication [S],” 1997.

[3] International Organization for Standardisation (ISO), “JTX 1/SC17.ISO/IEC 7816 Identification Cards-Integrated Circuit(s) Cards with Contacts [S]”.

[4] Y.-L. Wei, H. Zhu and B. Qiu, “Authentication Technology Research of Information Safety Based Dual Factor,” *Journal of Shandong University*, Vol. 40, No. 3, 2005.

[5] V. Chopra and J. Eaves, “Jsp Programming,” Posts and Telecom Press, Beijing, 1999.

[6] M.-H. Xu, “Java Web Integrate Development and Project Design,” Posts and Telecom Press, Beijing, 2010.

