"HENRI COANDA"
AIR FORCE ACADEMY
ROMANIA

"GENERAL M.R. STEFANIK"
ARMED FORCES ACADEMY
SLOVAK REPUBLIC

INTERNATIONAL CONFERENCE of SCIENTIFIC PAPER
AFASES 2014
Brasov, 22-24 May 2014

USER AUTHENTICATION TO A WEB SITE USING FINGERPRINTS

Mihai Lica PURA

Faculty of Military Electronic and Information Systems, Military Technical Academy, Bucharest, Romania

Abstract: *Nowadays, authenticated access to web sites becomes more and more important. Usernames and passwords are one of the easiest ways to accomplish it. But a more secure approach is to use biometrics. This article presents a very simple modality to use fingerprints to control the access to a web site. The client component of the proposed architecture is an ActiveX control that communicates with the fingerprint reader and sends data back to the web server. The server component is an ISAPI Server that processes the requests of the client regarding authentication and does or does not grant access to the web site. The ActiveX is integrated in the web page with the help of Java Script.*

Keywords: *web site authentication, biometrics, fingerprint*

1. INTRODUCTION

Since its creation, Word Wide Web (WWW) has continued to grow and to impact the everyday life of people ([1]). News, mail, banking, storage, and so on, are all provided as web services through some web sites. But such important services can be used with confidence only if they also provide the necessary security. The five most important security objectives are authentication, confidentiality, integrity, non-repudiation, and availability. For the web servers and for the communication between the client applications and the web servers, all these objectives (web server authentication by the user, confidentiality and integrity of the exchanged data, availability of the server) were addressed and accomplished mainly by using Public Key Infrastructure (PKI) and cryptographic operations through protocols like TLS (Trusted Layer Security) or SSL (Secure Sockets Layer) ([2]). But so far, authentication of the users is still mostly based on credentials of type username and password.

While this type of authentication can assure the needed confidence, still it has some disadvantages that can drastically diminish the provided security. In this article we present the use of biometrics to provide authentication to critical web sites like the one described above, with the purpose of eliminating the disadvantages of using passwords. The idea is not new. But the solutions that can be found are only a few ([3, 4, 5]), and all of them are proprietary solutions. This means that they are not free, and that they have to be integrated in the web site that will use them, which is not always easy to do. So we have proposed to develop an open source web site authentication solution based on fingerprints, and to use it to provide authentication add-ons to the most common CMSs (Content Management Framework) that are used for publishing web

content, like Joomla, Silverstipe and Drupal, in order to simplify their use and to ease their spreading.

The rest of the paper is organized as follows. In the second section we present the disadvantages of using password based authentication, the advantages of using biometrics and the architecture of our authentication solution. The third section gives some implementation details for the components of our solution. The last section contains the conclusions and our future work directions.

2. THE ARCHITECTURE OF THE AUTHENTICATION SYSTEM

We will start by presenting the disadvantages of using username/password pairs for user authentication. The first obvious disadvantage is the fact that the password has to be remembered by the user, because writing it down would be insecure, so the users tend to choose easy to remember strings of characters that can be guessed by attackers or can be subject to dictionary attacks. One solution to this problem would be the use of password management applications ([6]), but few people have the necessary knowledge and training to manage and use this kind of software. Also, many of such applications have been proved error prone, so using them is not necessary the best choice.

The second disadvantage of using usernames and passwords is the fact that any person who knows the username and the password of a user can authenticate to the web server. For the legitimate user it is difficult to realize that someone else has the password, until unwanted operations are already performed by the malicious part, when, of course, it is too late. Countermeasures to this issue exist. They consist in enforcing a password change policy by the web server, and/or the use of user activity logs. Changing the password on a regular basis will assure that if a malicious part has entered in the possession of a password, it will not have time to use it. Also, by consulting the activity log for his own account, the legitimate user can see if someone else has successfully authenticated to the server from a different location, or at a different moment of time. Of course, this means that the legitimate user has to keep track of the IP she uses, and of the times it has authenticated to the server. But most of the users are not trained to perform these actions, or they neglect them because of the extra operations involved.

The use of biometrics in general and of fingerprints in particular addresses both of these problems. First of all, the user does not have to remember anything that is to be kept secret. A username is still being used, but it is public information. Second, only the legitimate user will be authenticated to the web server, because no one else can have her biometric data.

We will now continue with the presentation of our proposed authentication framework. In a typical scenario for accessing a web site there are two components: the web server and the client application. The client component is usually a web browser and it runs on the client machine, which is the personal computer of any user who wants to access the web site. The server component runs on a remote machine and answers the client’s requests. For an authenticated access to a web site, the client, prior to accessing the web pages served by the server, needs to transmit to the server a secret only knew by the two of them, that uniquely identifies her. Typically, this secret is a username-password pair. Based on the client’s identity established in this way by the web server, the server knows what pages the client is entitled to request.

The password is entered by the client in a text box of the sign up page and then transmitted to the server as any other information from a web form. But when using fingerprints, we need one more component. That is because the fingerprint reader has to be accessed by the web page to get the information it generates about the fingerprint that it has scanned. This component has to run on the client machine, because the fingerprint reader is attached to it, but also needs to communicate with the server to send the credentials of the client who wants to authenticate.

In conclusion, the proposed authentication framework has three components: the web server, the web page that runs in a browser and the client application accessing the fingerprints reader. We will present all these components in the following sections.

2.1 The ActiveX client component. The technology that we have chosen to create the client side component responsible of accessing the fingerprints reader is ActiveX. Presenting the ActiveX technology and the development of ActiveX components is out of the scope of this paper. For more information on this please see ([7]).

We have defined a single method for our ActiveX component. This method displays a modal window. The window has an edit box component for entering the username, a label for displaying the error or the success messages and three buttons. One of the buttons is for exiting. The other two buttons correspond to the two actions necessary for a biometric authentication. The first step is the enrollment of the user. For this we have settled the *Enroll* button. This operation is only required to be taken one time, and one time only. That is at the registration of the user. Then, every time the user wants to authenticate, he has to use the *Verify* button.

In the enroll operation, the fingerprints reader gets a template of the user's fingerprint that will be store in the database of the server. Then, each time the authentication process takes place, the fingerprint taken by the reader is verified against the template stored in the database. If they match, the user is authenticated with the entered username. If they do not match, the verification fails. After performing verification, the modal window returns 0 or 1, depending on the success of the operation (0 if verification failed and 1 if it was successful). The script that uses this ActiveX will use the result to grant access to the web site or not.

2.2 The web page.

To use this ActiveX component, it has to be placed on the login page of the target web site. This is done by using the HTML tag *object*.

Here is an example:

```
<object
    id="BioAuthCtrl"          name="BC"
classid="clsid:9F1EAE3D-C2E8-4171-BB80-
D62B91F5B0FE"          height="300"
VIEWASTEXT style="width: 530px">
</object>
```

After this, the object can be used in the JavaScript from the page through its name, "*BC*" in our example.

Let us presume that we have associated a function *f()* with the *onclick* event for a sign up button on a login web page of a web site. To use our ActiveX for fingerprint authentication, the JavaScript code would be:

```
function f(){
    var x=BC.enroll();
    if(x == 1)
        // grant access to the site
    …
    if(x==0)
        // forbid access to site
        …
}
```

2.3 The server component. The server side component was build using ISAPI Server technology (Internet Server Application Programming Interface). The communication between the two components was implemented using *CGatwayInterface* implementation of the *WinInet API* provided in

[8]. Presenting the ISAPI Server technology and its programming is out of the scope of this paper. For further information please consult [9].

The ISAPI Server we have implemented can process the following three requests: *storeTemplate*, *saveTemp* and *performVerify*.

The prototypes of the three requests are provided below, using a hypothetical example for the URL of the target web site:

http://my.site.com/BioAuthServer.dll?saveTemp&template="string_template_part1"&username="string_username"

http://my.site.com/BioAuthServer.dll?performVerify&template="string_template_part2"&username="string_username"

http://my.site.com/BioAuthServer.dll?storeTemplate&template="string_template"&username="string_username"

The *storeTemplate* request saves the template of the user's fingerprint in the database of the server, associating it with the corresponding username, also sent by the client. It is used in the user enrollment phase.

The other two requests, *saveTemp* and *performVerify*, are used in the user authentication phase. The verification process has been broken into two parts. This was necessary because of the maximum data length that can be transmitted in a *WinInet* communication, which is 1024 kilobytes. The fingerprint's data necessary for verification against the template exceed this maximum size. That is why the data are sent in two parts: *saveTemp* request sends the first 1024 kilobytes of the fingerprint's data, and *performVerify* sends the last kilobytes of it. When handling a *performVerify* request, the web server concatenates the two parts of the data of the fingerprint and performs verification. It then answers to the client telling it if the verification was successful or not.

3. IMPLEMENTATION DETAILS

To simplify our implementation, the database of the users' fingerprints templates was constructed as a directory of files. At the enrollment phase, a file is created for each user, using the corresponding username sent by the client. This file contains the template of the fingerprint received from the fingerprint reader.

When the *saveTemp* request is performed, the ISAPI Server saves the first part of the fingerprint's template as a temporary file. If *storeTemplate* request follows, the final template file is build from the temporary file and the second part of the template. When *performVerify* request is performed, the username passed to the server is used to search for the file that stores the respective template. If the file does not exist, the verification fails. If the file exists, it is opened and the template is read from it. Then the template received from the client is verified against the template stored by the web server. If they match, the verification is successful. Otherwise, the verification fails.

One thing that we have to have in mind is that these three requests are in form of strings. This means that the templates (that are binary data) are first serialized by the client ActiveX and only after that they are sent to the server as HTTP requests. At the server side the deserialization takes place, obtaining the binary data.

To protect the data exchanged between the client and the web server, the serialized fingerprints data are first encrypted with a session key previously computed by the two parts. To eliminate the possibility of reply attacks, the salted encryption is used. Of course, at the server side decryption is performed prior to any other processing. Because of the fact that encrypted data is binary data, it is encoded as a BASE64 string. The use of HTTPS is also highly recommended.

The ActiveX component needs to have access to the local resources of the computer the web browser is running on. Only this way it will be able to communicate with the fingerprints reader. To authenticate the ActiveX component to the user, it will be digitally signed. This way the user will be able to check if the application that tries to access

the fingerprints reader is really the one used in the web server authentication process.

So far we did not mention anything about the biometric specific functions. We will skip this information because these functions are highly dependent to the fingerprint reader. But, generally, these functions are part of an API that comes together with the driver of the reader. The important thing is that it is very easy to configure the ActiveX component to use the fingerprints reader desired by the user.

4. CONCLUSIONS

In this paper we have presented a web site authentication framework based on fingerprints. We have described the need for an alternative to classical password based authentication, at least regarding critical web sites like the ones used for e-banking, and e-mail. We also highlighted the advantages of using biometric credentials in general, and fingerprints in particular.

The presented framework has some drawbacks that we intend to solve in the near future. The most obvious one is the fact that the template's database is primitive. This aspect can be improved by using a relational database server, like Microsoft SQL Server, or MySQL. Regarding human-computer interaction, the ActiveX window must be split in two separate windows: one for the enrollment phase, and one for the verification phase. The two operations have total separate flows, so the separation of the windows is necessary.

Another possible improvement is the replacement of the ISAPI Server with an ISAPI Filter. This way the configuration of the authentication process at the server level will be easier to make.

After addressing all these aspects, we will develop fingerprint authentication add-ons for the most used web content CMSs, like Joomla, Silverstripe and Drupal, and we will make them available on the corresponding web pages of these tools. This way, fingerprint authentication will be easy to add to any web site created with one of them.

We would like to thank to engineer Florin Panta who participated to the implementation of the presented framework by the work he has done for his diploma paper.

REFERENCES

1. Tim Berners-Lee, *Information Management: A Proposal*, CERN, 1989, [online]. Available: http://info.cern.ch/Proposal.html (February 2014).
2. The Transport Layer Security (TLS) Protocol Version 1.2 RFC 5246, [online]. Available: http://datatracker.ietf.org/doc/rfc5246/ (February 2014).
3. BIO-Key, WEB-KEY, [online]. Available: http://www.bio-key.com/products/overview-2/web-key (February 2014).
4. Plurilock Security Solutions Inc., FINGERPASS, [online]. Available: http://plurilock.com/products/fingerpass (February 2014).
5. DigitalPersona, DIGITALPERSONA ONLINE, [online]. Available: http://www.digitalpersona.com/Authentication-Solutions/DigitalPersona-Online/DigitalPersona-Online/ (February 2014).

6. Gaw, Shirley, Edward W. Felten, *Password management strategies for online accounts*, Proceedings of the second symposium on Usable privacy and security, ACM, 2006.
7. Kedar Gore, *A Complete Web ActiveX Control*, [online]. Available: http://geekduo.com/geekduo/2011/11/a-complete-web-activex-control-part-i/ (February 2014).
8. Steve Zimmerman, *Designing ActiveX Components Part II: Implementing Internet Communication with WinInet*, Microsoft Interactive Developer, 1997.
9. The ISAPI Developer's Site, [online]. Available: http://www.genusa.com/isapi/isapitut.htm (March 2014).