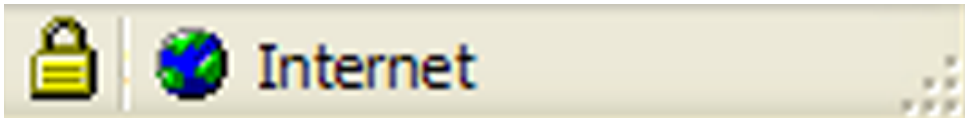
Integrated Windows Authentication

Integrated Windows Authentication

by Matthew D. Langston

Introduction

Many users are familiar with SSL encrypted web pages that ask them for their username and password to login to web sites. With SSL, web browsers use a gold lock as a visual cue to indicate to users that their username and password will be transmitted securely over the Internet. For example, this is what Wells Fargo Bank customers see in the lower right hand corner of Internet Explorer 6.0 and FireFox 1.0 when they log into their account:

Internet Explorer 6.0	FireFox 1.0
	

Although SSL is widely used to allow users to securely login to web sites, it is not the only method that modern browsers support. Another method, which is at least as secure as SSL (if not more so), is called *Integrated Windows Authentication* (hereafter called *IWA*). Most web browsers (all versions of Internet Explorer, and recent versions of Gecko-based browsers such as FireFox 1.0) support IWA.

In some ways IWA is more secure than SSL since IWA never sends the username and password to the remote web server. Although SSL sends the username and password in an encrypted format, once it arrives at the web server it is in clear-text and could be accidentally exposed by an inexperienced web programmer. IWA does not suffer from this vulnerability since the username and password never leave the user's browser.

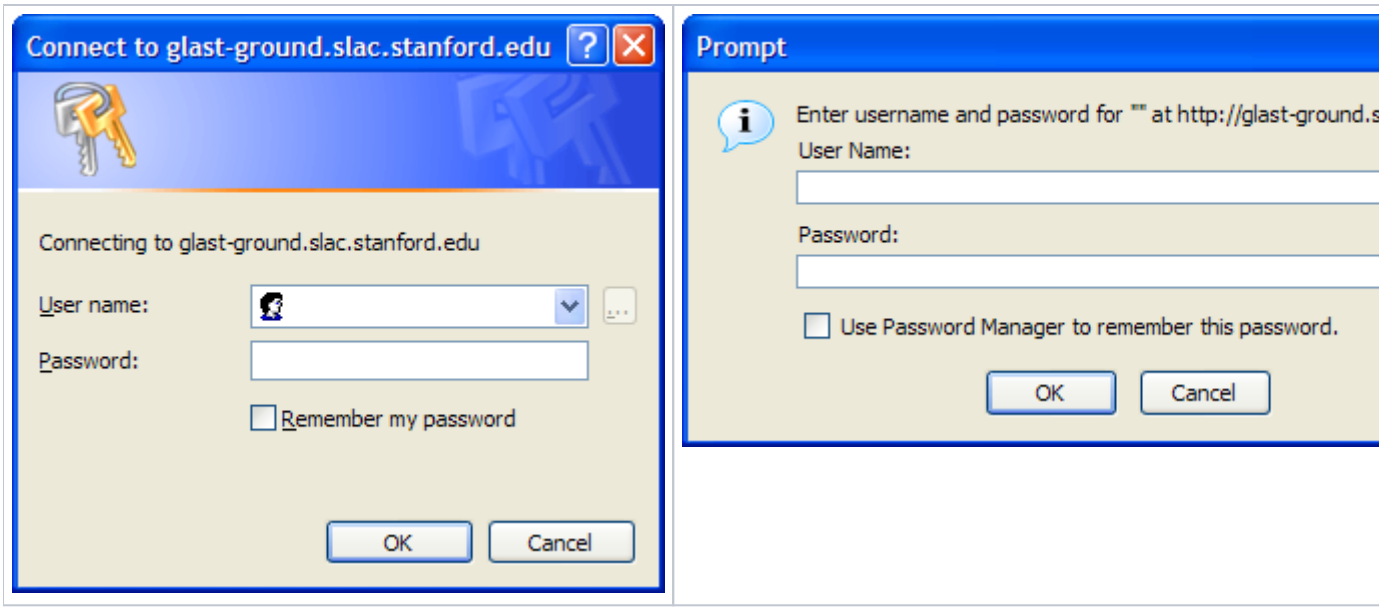
SSL uses the widely recognized *gold lock* visual cue to indicate to the user *it is safe to type your password*. IWA uses a different (but just as valid) visual cue to reassure the user *it is safe to type your password*. Since the visual cues are different for the SSL and IWA methods, some reassurance of the safety and validity of IWA is being provided to the GLAST community in the form of this article.

How IWA works

Roughly speaking, there are two ways to authenticate a user to a web site called **Forms Based Authentication** and **Browser Based Authentication**. The method many users are familiar with is *Forms Based Authentication*, which is when a form embedded in a web page prompts a user for their username and password over an SSL connection to the web server. The user types their username and password into the web form and clicks the submit button which sends the credentials to the web server over the encrypted SSL channel for authentication. It is important to point out that *the user's web browser has no idea that the user is logging into the web site* - all that the web browser knows is that it is sending information to the remote web site over an SSL channel.

The *Browser Based Authentication* mechanism is different in that it uses the browser's built-in functionality to authenticate a user to a web site. It is important to point out that *the user's web browser fully participates in logging the user into the web site* - this is a completely different approach to the Forms/SSL method. Since the browser knows it is logging the user into a remote web site, it can use a built-in dialog box to ask the user for their username and password. Here are the dialog boxes used by Internet Explorer 6.0 and FireFox 1.0:

Internet Explorer 6.0	FireFox 1.0



For both the *Forms/SSL* and *Browser* based authentication mechanisms, it is important that the user trusts the web site they are logging into. For example, just because you use an SSL encrypted form to send your username and password to a remote web site doesn't mean that your password is safe. If the programmer who created the remote web site is inexperienced in security issues, they could easily do something to compromise your password without intending to do so. Because security is so important and so easy for a programmer to get wrong, the Department of Energy requires SLAC to not allow programmers to ever ask a user for their username and password unless they obtain special permission from the lab.

IWA is an example of *Browser Based Authentication* since it is a feature that must be built-in to the browser. As with *Forms/SSL*, the user must trust the web site they are sending their credentials to. Since http://glast-ground.slac.stanford.edu/ is an official GLAST web site that has been vetted by SLAC Computing Services (SCS), GLAST users can trust that it is safe and secure to provide their SLAC credentials to the web site. In the dialog boxes above, the visual cue that it is safe for the user to enter their username and password into the dialog box is the HTTP address in the dialog box. It is clear to the user that they are connecting to the web site http://glast-ground.slac.stanford.edu/, and since they trust this web site they can safely enter their username and password.

Under the Covers of IWA

For those of you interested in the details of IWA, I'll walk you through the HTTP headers of a web browser connecting to http://glast-ground.slac.stanford.edu/ so that you can see how the cryptographic exchange works. In each of the following diagrams, the HTTP header sent by the browser to the remote web server is shown first, followed by the remote web server's response back to the browser.

Unauthorized User Visits Web Site

http://glast-ground.slac.stanford.edu/

```
GET / HTTP/1.1
Host: glast-ground.slac.stanford.edu
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.7.5) Gecko/20041107 Firefox/1.0
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,/;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Cookie: CFTOKEN=84811003; CFID=906

HTTP/1.x 401 Unauthorized
Content-Length: 1656
Content-Type: text/html
Server: Microsoft-IIS/6.0
WWW-Authenticate: NTLM
Date: Sun, 19 Dec 2004 01:23:45 GMT
```

Note that the web server responds that the browser it is not authorized to access the web server (the HTTP/1.x 401 Unauthorized tells you this), and that the only valid form of authentication that the web server will accept is IWS (which is what the WWW-Authenticate: NTLM line tells you). Since IWA is built-in to the browser (in this case FireFox 1.0), it prompts the user for their username and password.

A hash of these credentials (**not the credentials themselves**) is passed to the web server (the line Authorization: NTLM TlRMTVNTUAABAAA... in the listing below), which allows the web server to look up the user in the Windows password database and to construct a unique encrypted challenge that the browser can only decrypt with the user's unique password. The long line of characters sent by the web server to the browser (after the WWW-Authenticate: NTLM in the listing below) is the encrypted challenge:

Web Server Challenges User

http://glast-ground.slac.stanford.edu/

```
GET / HTTP/1.1
Host: glast-ground.slac.stanford.edu
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.7.5) Gecko/20041107 Firefox/1.0
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,/;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Cookie: CFTOKEN=84811003; CFID=906
Authorization: NTLM TlRMTVNTUAABAA...

HTTP/1.x 401 Unauthorized
Content-Length: 1539
Content-Type: text/html
Server: Microsoft-IIS/6.0
WWW-Authenticate: NTLM
TlRMTVNTUAACAAAA...
Date: Sun, 19 Dec 2004 01:24:06 GMT
```

Back at the browser, the browser attempts to decrypt the challenge with the user's password to get the answer to the challenge, which the browser then sends to the web server as proof that the user is who they claim to be. In the listing below, the string of characters after the line Authorization: NTLM is what the browser thinks the answer is. In this case, the user provided valid SLAC credentials to the browser, and the original page is served:

Browser Correctly Answers the Challenge and Web Server Sends Original Page

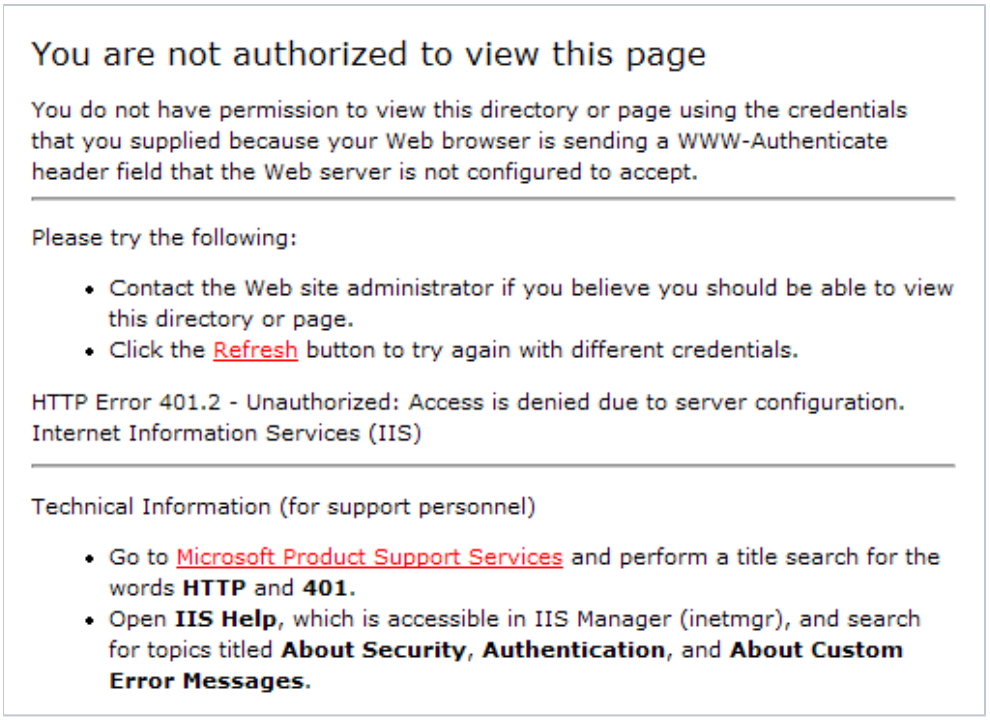
http://glast-ground.slac.stanford.edu/

```
GET / HTTP/1.1
Host: glast-ground.slac.stanford.edu
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.7.5) Gecko/20041107 Firefox/1.0
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,/;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Cookie: CFTOKEN=84811003; CFID=906
Authorization: NTLM TlRMTVNTUAADAAAAGA...

HTTP/1.x 200 OK
Connection: close
Date: Sun, 19 Dec 2004 01:24:08 GMT
Server: Microsoft-IIS/6.0
Set-Cookie: JSESSIONID=98307ef1b78b$3F$B77B;path=/
Set-Cookie: CFAUTHORIZATION_glast_ground=;expires=Fri, 19-Dec-2003 01:24:08 GMT;path=/
Content-Type: text/html; charset=UTF-8
```

If the web server determines that the answer is correct, the web server sends the originally requested page to the browser. If not, the user is presented with an "Unauthorized Access" page. If given the wrong username and/or password, the user will see the following page.

If the User Provides the Wrong Password



A new challenge is presented to the user for every request, which prevents a hacker from assuming the identity of the user to request other pages that the user hasn't authenticated to yet.

Conclusion

IWA is a valid and secure way for web sites to authenticate users over insecure networks such as the internet. The username and password are never sent over the network - they are held by the browser and used to answer challenges from the remote web server. It is just as secure as SSL (if not more so), and uses visual cues to indicate to the user who is requesting their credentials. If the user trusts the web site, then they can feel comfortable submitting their credentials to it. A new challenge/resposne is exchanged between the remoet web server and browser for every new request.

For GLAST users, since SCS has vetted the web site http://glast-ground.slac.stanford.edu/, they can trust that security is implemented properly and should feel secure in using it.