

# Chicken and egg issue with Cookie based login?

*Source:* <http://www.derkeiler.com/Newsgroups/comp.security.misc/2005-04/0110.html>

---

*From:* Julio ([julio\\_at\\_lalaland.com](mailto:julio_at_lalaland.com))

*Date:* 04/06/05

Date: Wed, 6 Apr 2005 04:39:25 -0400

I have few questions I hope someone can clear up for me with the cookie security discussion at this W3C document:

<http://www.w3.org/Security/Faq/wwwsf2.html#CLT-Q10>

I'm trying to implement a cookie based authentication system for our private web server. The document suggested a popular algorithm/method to generate a MAC.

```
MAC = MD5("secret key " +  
    MD5("session ID" + "issue date" +  
        "expiration time" + "IP address" +  
        "secret key")  
    )
```

It also says this about the secret key:

"This algorithm first performs a string concatenation of all the data fields in the cookie, then adds to it a secret string known only to the Web server."

My questions are:

How is this a "secret" only know to the server if its needs to be passed to the client in order to generate the hashed cookie? Sounds like a chicken and egg scenario, since you need to contact the server to get the secret key in the first place?

Second, would be an example of the "Session ID" or more general, what is an example for a secret key and session id for a system that users a username/password database on the server side?

if my web server has a built-in user database storing a user name and password, and the typical login form for a web page would be for a userid/name and password prompt, it seems to me that this "Session ID" could be the combo of the userid/password? or....

## comp.security.misc: Chicken and egg issue with Cookie based login?

Since this is a hash, there needs to be some unhashed value passed to the server, like the user name and only include the password (Session ID?) in the hash in order to do the username lookup and perform a hash comparison with the password?

Is the "Secret Key" the user's password and the Session ID the user's name?

Finally, what I came up with (to explore) is this:

$$\text{MAC} = \text{MD5}(\text{password} + \text{MD5}(\text{userid} + \text{sessionkey} + \text{password}))$$

where sessionkey is unique hash key for the new login session. It contains the following:

$$\text{sessionkey} = \text{MD5}(\text{IP} + \text{unique counter} + \text{timeout window})$$

All 3 parameters are cached at the server mapped to the sessionkey.

Next the cookie is set:

$$\text{COOKIE} = \text{username} + ":" + \text{sessionkey} + ":" + \text{MAC}$$

At the server, the username is looked up to get the user's database password, and the sessionkey is looked up to get the 3 session parameters. The appropriate checks are then performed; the IP is compared; the timeout window is checked, and the MAC is regenerated based all the server ready variables.

I know its not 100% fool proof, but does this make sense?

Currently our web server supports BASIC and DIGEST (and SSL), so of course, if anyone is concern with security, they are steered to enforcing SSL or atleast DIGEST for non-SSL operations. The cookie support has been a big request mainly to offer the complete log off capability lacking in standard BASIC/DIGEST methods.

So I want something with cookies that is atleast better than plain text BASIC using a similar "session key" challenge concept as done with DIGEST.

Make sense?

Thanks