My Funny Val()entine and SQLi

Posted At : June 15, 2009 2:30 PM | Posted By : Mark Kruger
Related Categories: ColdFusion, Coldfusion Security

Regular readers know I'm always on the lookout for interesting issues regarding SQL Injection and ColdFusion. This year has been a banner year for injection on ColdFusion sites and if you are not on the Cfqueryparam bandwagon yet I have one more example of a code that might *seem* to be inoculated but is not. It has to do with the use of val( )....

Example

I recently saw code like this:

```
<cfif isDefined('url.userid') AND val(url.userid)>
    <cfquery name="user" datasource="#mydsn#">
        SELECT     *
        FROM    users
        WHERE    userid = #url.userid#
    </cfquery>
</cfif>
```

Now this approach has been around for a very long time and it *seems safe* - but is it? Let's take a closer look at that IF statement. First we have:

isDefined('url.userid')

That's easy enough right? The code checks to see if url.userid actually exists. So far so good. The rest of it looks like this:

isDefined('url.userid') *AND val(url.userID)*

The programmer is taking advantage of a built in order of precedence inside of an IF statement. ColdFusion runs the "isDefined()" check first and returns false if it fails. If it doesn't fail that means the param *does* exist and it's ok to run the val function against it directly. Again, so far so good. Now let's talk about that Val() function.

Val() and Number Magic

In my observation of the common usages of val() I would have to say that a simple definition (based on usage) might be, "A function that takes any primitive variable and returns it back "as is" if it is a number, otherwise it returns zero (0)." That's right isn't it? Certainly the author of the code above thinks it is. He's taking advantage of CF's ability to treat any number as boolean for "true" and a zero as "false" so AND Val(url.userid) triggers true if the user has passed a number. Since the code only allows numbers to be passed in as url.userid, then the query is protected against SQL Injection by virtue of the fact that a non-numeric value cannot make it through - right? The problem is that there is a bit more to the definition than just "is a number" or "is not a number" - otherwise we would just use "isNumeric()".

What Does Val() Do

Here's the definition of Val() from the ColdFusion docs: *"Converts numeric characters that occur at the beginning of a string to a number."* Uh oh.... Val() takes an argument

as a *string*, goes through it in order and returns whatever number can be converted from that string by starting with the first character. As soon as it hits a non-number it stops and returns whatever it has so far. If the first character in the string is a number (other than zero) then this check will *always return true*. If you pass it a 1.0 it will return 1.0. If you pass it a 3abc, *it will return a 3* etc.

Why is that important? Take a closer look at the code above. If I were clever I would pass in userID=1 OR 1 = 1 and the query would most certainly return all of the records in the users database. So even though it might *seem* like this code is checking for a number it is actually of little use in this case. Any valid SQLi string would start with a number anyway.

The Fix

The use of CFQUERYPARAM would inoculate this query effectively. Oddly enough, given this post, the use of val() in the WHERE clause would also do the trick (as in WHERE userID = #val(url.userid)#).