SpoofingBug

A serious bug was discovered in versions of Shibboleth prior to 1.3e that results in clients having the ability to spoof the values of attributes supplied through CGI variables to applications whenever a particular header variable was left unset by the !SP software. A security advisory describing this problem is posted on the Shibboleth web site.

This topic outlines the bug's cause, discusses which deployments are most affected, and outlines the limitations of the fix present in the latest version. I encourage deployers to thoroughly explore the problem, attempt to reproduce the bug in newer versions, and post any concerns arising with their development tools.

Once more testing is complete, the 1.3e patch will be finalized, but the most current code is now available.

Background

The cause of the bug is the many-to-one mapping of header names to CGI variable names due to upcasing and replacement of some separator characters with underscores. It's exacerbated by the fact that different web servers use different rules, particularly with regard to how non-alphanumeric characters are handled. Some are turned to underscores, and some are left alone, resulting in strange or even technically invalid CGI variable names.

The unpredictability makes it difficult to prevent a client from sending a creatively malformed header that will map to an expected CGI variable reserved by an application for a particular user attribute. The techniques used to "clear" client-sent headers that might conflict were inadequate.

This problem is not confined to LazySessions because it's always possible for most attributes to be optional or suppressed at the !IdP, and many kinds of errors can result in valid sessions that appear to be missing attributes. It is, however, easier to reproduce with LazySessions because even the reserved variables like HTTP_SHIB_IDENTITY_PROVIDER can be spoofed.

It's also exacerbated and in some ways actually **caused** by punctuation. If your headers are all confined to one word names (or at least all jammed together), there are at least some potential spoofing avenues that will be prevented even with older versions. But not all.

The basic rule of thumb is that any time the Shibboleth software supplies a value for a CGI header, you don't have to worry about spoofing. It's when a value *isn't* supplied by Shibboleth that the variable became vulnerable. As I'll describe below, this is a simplification, particularly when not using Apache, but it serves as a basic guideline.

Finally, I believe that version 1.3e contains a relatively sound fix for the bug. I say relatively because I think there's a better fix possible (suggested by Peter Watkins), but with additional work. In the case of IIS, a lot of work. So I'm releasing a fix that I think works until more time can be spent on the problem. More on this below.

Apache Behavior

Apache seems to be the most sound CGI implementation of the supported platforms. It has a basic property across all extant versions (1.3, 2.0, 2.2) that any time two or more request headers transform into the same variable name, only the last header set/seen is used as the value. Multiple headers are **not** combined by Apache itself. This is really good, because Shibboleth always has a chance to set a value after the client.

Apache also has a very simple, consistent mapping rule. Any non-alphanumeric character becomes an underscore, the rest are upcased. Very easy, though also very simple to spoof as a result.

Working on these assumptions based on the code and testing, a simple fix was applied: when a header is "cleared", it is actually "set" with an empty value. If the client supplies its own value, the client loses. Shibboleth runs this operation on all headers it may be asked to set on all requests with the proper !AuthType. The result is a consistent fix that appears to function safely and uniformly for all development tools, CGI, !FastCGI, etc.

> *PLEASE report **any** Apache version or variant that does not exhibit this property to the Shibboleth team. This is not intended to be the only permanent fix, but if it's not even a temporary fix, we need to know. But I think it works.*

IIS Behavior

IIS is both good and bad news at the same time. Good, because it turns out that the only way to even **try** to clear a header from an IIS filter is actually to set an empty value. Bad, because it is utterly undocumented at everything it does with headers, and it does some **weird** stuff.

The upside is that strictly speaking, nothing was "fixed". The existing behavior appears to prevent spoofing attacks in most typical cases. I say most because there's an exception, and a qualification. The qualification is that I don't know **why** it works. Because there's no source code, and it doesn't provide a reliable way to examine the full header list, I'm forced to assume that because my testing with ASP indicates that setting an empty header suppresses a like-named CGI variable from the client that this is consistent behavior. I can't prove that, I just can't break it yet, at least on the versions I've tested. Again, I need more testers.

Now, the exception. CGI itself is broken without the 1.3e patch and a new, optional feature I added. ASP and similar scripting tools are not CGI-based tools. They run using more direct connections with the server and receive raw data that they expose to scripts. Actual CGI scripts, such as Perl, do **NOT** work right when a spoofed header comes in, even if an empty header is set (as long as the spoofed header doesn't have the exact same name as the Shibboleth header, of course). The spoofed value **WILL** be seen.

As I said earlier, it turns out that without a great deal of work, and possibly not even then, a filter cannot get a complete list of the headers sent by the client in order to protect itself and identify possible spoofing. Given that, I wanted a shorter term fix for any CGI applications out there using IIS.

My fix was to add a new `unsetHeaderValue` option to the ShibbolethXml `<Local>` element. It's a string valued attribute that can contain a special fixed token value to insert into any header that Shibboleth does **not** set a real value for, instead of using an empty string. Any CGI variable that's left unset with a real value will just contain the special value you choose. Obviously, it defaults to an empty string, so it works consistently with the older behavior.

My testing suggests that an actual value (as opposed to no value) will override a client-supplied value for CGI scripts. It's an ugly work-around, but appears to be the only one short of figuring out how to check every header for possible spoofing. I also would like to see a lot more testing before I conclude that CGI is safe. I would say it's an open question.

A final note on IIS weirdness. Unlike Apache, all kinds of weird header names can be sent and often end up output as variables without even colliding. In Apache "Foo.Bar.Baz" will spoof "Foo-Bar-Baz", but on at least some IIS versions, it actually creates a unique variable named "HTTP_FOO.BAR.BAZ". Ridiculous. More importantly, undocumented.

Sun/iPlanet

This isn't getting much use, but for completeness, the NSAPI module seems to be somewhat similar to IIS, though not quite as weird and certainly more capable. It was vulnerable to spoofing because I used an "unset" technique similar to Apache, and I applied a similar fix for now, just setting an empty header. Then I replace the empty header with a real one later when a value has to be set. As with IIS, the empty value appears to reliably block a client-supplied value for tools like Cold Fusion.

CGI needs more testing, but I think it likely to be broken as with IIS, and the `unsetHeaderValue` option will probably be needed here as well. I will update as I learn more.

Summary

Prior to version 1.3e:

The only apparently safe deployments are IIS using ASP or probably other in-proc non-CGI scripting tools. It could well be that not all IIS or ASP versions and configurations are safe, so I suggest testing for obvious problems.

In theory, it's also possible to claim that when LazySessions are not used and you know that all possible attributes will have values all the time, you're safe. But the latter is almost impossible to actually guarantee. At a minimum, you'd need to force failed queries to abort a new session by setting `propagateErrors` to `true` , which is not the default anymore.

With the 1.3e patch:

Apache CGI seems to be safe, but non-CGI tools like PHP need further testing.

IIS should be the same as before except that CGI **MAY** be usable when the `unsetHeaderValue` option is used to insert a dummy value into any potentially empty variables. Applications of course need to know about the dummy value chosen so they don't misinterpret it as a real value.

Sun/iPlanet should be safe, but as with IIS, the new option will be needed for CGI-based application safety.

If after reading this, you come to the conclusion that this all sounds fishy, you're not alone. I believe that as Peter Watkins suggested, the best fix is to eat some CPU and actually search the entire header list for any potential spoofing. This should be simple to implement, except for IIS, so that's the top concern.