

[NT] ASPSession ID's Vulnerability

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2001-12/0094.html>

From: support@securiteam.com

Date: 12/22/01

From: support@securiteam.com

To: list@securiteam.com

Date: Sat, 22 Dec 2001 06:41:04 +0100 (CET)

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: <http://www.securiteam.com>

-- promotion

When was the last time you checked your server's security?

How about a monthly report?

<http://www.AutomatedScanning.com> -- Know that you're safe.

ASPSession ID's Vulnerability

SUMMARY

A potential security risk has come to light for internet applications that make use of ASP SessionID's to maintain user state information, even if most of the site is over a secure SSL session.

DETAILS

This vulnerability is caused by two flaws in the IIS architecture it is possible that:

(a) An ASP SessionID is allocated even if the .ASP pages sets the session state to "false" by <% @ EnableSessionState=False %>. See Knowledgebase article <<http://support.microsoft.com/default.aspx?scid=kb;EN-US;q273974>> Q273974 for more details.

In addition

(b) The same ASP SessionID is shared between a HTTP and HTTPS (SSL) session. See Knowledgebase article

<<http://www.microsoft.com/technet/support/kb.asp?ID=274149>> Q274149 for more details. This is also discussed in MS Security Bulletin

<<http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS00-080.asp>> MS00-080.

Securiteam: [NT] ASPSession ID's Vulnerability

The security problem:

What does this mean? It means that you might be sending out ASPSessionID's when you think you are not, and that the same ASP SessionID cookie is being sent in clear text HTTP as is used in the encrypted HTTPS (SSL) session.

So, if we can get that ASPSessionID cookie we can impersonate another user even over an SSL session. If a new user connects the server will just seamlessly recognize the SSL session, and when they then present the valid ASP SessionID the application will then serve them up another users session (e.g. for an online bank: their account details...).

The normal assumption here would probably be that a user would have to have malicious intent to "sniff the wire" (intercept network traffic) to get the ASP SessionID cookie and then present it back to the host web server. Unfortunately, in this case it is not true.

A poorly configured web cache or proxy server might just do the hacking for you by caching either the initial Set-Cookie header or the cookie itself. Thus, the second user who requests the same URL might get the cookie inadvertently sent to them out of the cache, and on subsequent requests might retrieve information from another user's session. (See in the URL in the additional information section below for further reading on cookie caching).

How can we tell if this is happening? Look at your IIS logs and scan for duplicate Source IP/ASP SessionID pairs. If the source IP's are significantly different, i.e. enough to discount that they are just going through load-balanced cache/proxies then it might be happening to your site/application.

Resolution(s):

What can we do to avoid this happening?

(a) Explicitly turn OFF ASP session handling in the IIS metabase for any virtual directories that don't need it, and don't rely on the `<%@ EnableSessionState=False %>` directive. This should be mandatory for any part of the site that is HTTP not HTTPS. Ideally, the site/application should be structured that the HTTP portion of the site is distinct from the HTTPS secure area, and the insecure part can be load-balanced across servers without any need for session persistence.

(b) Implement the patch and fix in Q274149/MS00-080 described above. Note that this requires more than just installing the patch – it requires that some changes be made to the metabase i.e.: "adsutil set w3svc/1/AspKeepSessionIDSecure 1". This change can be set on the site level, or scoped down to any IIS application namespace for finer grained control.

(c) Try to stop any of the pages being cached by setting the following either in the .ASP page or in the virtual directory custom headers

property

```
Response.CacheControl = "no-cache"
```

```
Response.AddHeader "Pragma", "no-cache"
```

```
Response.expires = -1
```

(d) Add a random value to the URL for each client when the client establishes a session – this unique URL would then also help to prevent caching. Obviously, this unique value should not be used for any session handling, as it may also be available 'en clear' for any HTTP part of the site.

(e) Don't use ASP SessionID's at all. Implement another, more robust, load balancer friendly user state mechanism. Note that this may change with Net/IIS6 distributed state mechanisms.

(f) Related to (e) use a more robust security system like Netegrity SiteMinder that is able to resist session cookie spoofing.

(g) Make the entire site HTTPS: – but this obviously incurs a performance hit, which can be partly overcome by the use of SSL acceleration.

ADDITIONAL INFORMATION

The following shows the ASP SessionID structure:
A typical ASP cookie from an IIS Log looks like this:

```
ASPSESSIONIDGQGGQYNO=LJALNFJCGLOICFEPIAPBFDEJ
```

There is a structure to it like this

```
ASPSESSIONID GQGGQYNO = LJALNFJCGLOICFEPIAPBFDEJ
```

The blocks have the following meanings:

ASPSESSIONID this is a constant

GQGGQYNO this is an 8-character "munge" of the process ID for the process running the application
= (this is an equals sign)

LJALNFJCGLOICFEPIAPBFDEJ is a 32 character "munge" of the 32 bit session ID (see later for how session ID is created)

Session ID is created from a random seed number that is generated when the system starts up). The random seed is incremented every time a new session starts. Note that the "munge" doesn't increment in the same way that the Session ID does.

Securiteam: [NT] ASPSession ID's Vulnerability

Since the 8 char string after ASPSESSIONID is a "munge" of the process ID it will be (a) the same for all "In-process" applications (b) a different value is shared for all "Medium isolation (pooled)" applications and (c) unique for each Out-of-process application.

Note to that for IIS5 the ASP session id COOKIE value is that same FOR THE ENTIRE WEBSITE... even though that ASP SessionID is DIFFERENT for each application name space (global.asa). Therefore, what you get back from session.sessionID will be different. For more information see <<http://www.microsoft.com/technet/prodtechnol/iis/reskit/iis50rg/iischp6.asp>> <http://www.microsoft.com/technet/prodtechnol/iis/reskit/iis50rg/iischp6.asp> (follow the "The Middle Tier" link and look for "ASP Session IDs Are Not Unique")

Cookie Caching:

See "point two"

<<http://security.royans.net/info/articles/loadbalancingproblems.shtml>>
<http://security.royans.net/info/articles/loadbalancingproblems.shtml>

Old list thread re cookie caching

<http://www.roads.lut.ac.uk/lists/http-caching/1996/02/0066.html>

Old AOL FAQ that also talks about their view of caching

<http://webmaster.info.aol.com/faq.html>

The HTTP 1.1 RFC that specifically tries to add new directives to deal with this issue HTTP 1.1

<<http://www.w3.org/Protocols/rfc2068/rfc2068.txt>> RFC2068

Other related MS software bugs:

Site Server GUID issue

<<http://support.microsoft.com/support/kb/articles/Q238/6/47.ASP>> Q238647

Another Site Server issue

<<http://support.microsoft.com/support/kb/articles/Q263/7/30.ASP>> Q263730

The information has been provided by <mailto:stephen_thair@YAHOO.CO.UK>
Stephen Thair.

=====

This bulletin is sent to members of the SecuriTeam mailing list.

To unsubscribe from the list, send mail with an empty subject line and body to:

list-unsubscribe@securiteam.com

In order to subscribe to the mailing list, simply forward this email to: list-subscribe@securiteam.com

=====
=====

DISCLAIMER:

The information in this bulletin is provided "AS IS" without warranty of any kind.

In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential,

Securiteam: [NT] ASPSession ID's Vulnerability

loss of business profits or special damages.

- **Previous message:** support@securiteam.com: "[EXPL] ATPHTTPd Buffer Overflow Exploit Code"
- **Messages sorted by:** [\[date \]](#) [\[thread \]](#) [\[subject \]](#) [\[author \]](#) [\[attachment \]](#)