The Progress Electronic Magazine

An Amduus™ Information Works, Inc. Publication

Table of Contents

Publisher's Statement

Have you ever had that annoying message “Internal Server Error” appear while your working code, and yet the message is not in the log files, nor are you able to get it on the web page?

That has happened to me a couple of times. It usually happens when it is in a complicated OUTPUT-HEADERS procedure that is trying to determine if a person has any business being on that page or what information the person is allowed to see on the page.

This issue provides a simple solution for this tricky problem on UNIX oriented operating systems (sorry Window's fans.)

I also apologize for not getting the code from the previous E-Zine in the web site as I said it would be found. I got busy and just totally forgot about it. Having to earn a living tends to do that now and then (knock on wood!) You can find it here: http://amduus.com/OpenSrc/SrcLib/OOP/

Note in the near future another little version is going to come out.

> *A little note to the unemployed – my old head hunter James Arnold is back in business scouring the marketplace for jobs and for people. If you want to let him know that your looking, zap him an email at this address: jarnold@mylinuxisp.com . He has done me right in the past landing me contracts in the $120,000 a year range. I don't know what the market is like these days (I am a lowly W2'er now), but I believe he will try to find a fair job for you.*
>
> Also... I am open sourcing Tresca (which was called Service Express.) If you want a copy of the latest build, let me know. It is still available for rent on Amduus' machines. See http://amduus.com/serviceexpress/PricePerUser.html for pricing.

Lets get on with the fun!

Scott Auge

Founder, Amduus Information Works, Inc.

Logging Full HTTP Messages

By Scott Auge

There are times when you want to see everything that comes into and out of your webspeed transaction server. Believe it or not, this is possible with some simple scripting on a UNIX based[1] platform.

There are a lot of things we do not see on the web page when an interaction is made with a web server. I shall go into this a bit more later once I have explained the script below.

This method only works on CGI based set-ups. Of course you can continue to use some kind of web server plug in if you wish in your main site, but your development site will need to be set up CGI for this kind of solution.

The wonderful thing about CGI is that it uses stdin and stdout to communicate to and from the web server and the webspeed transaction server. It is in this little pipe we

1 This code was tested on a Linux based computer. This scripting code will also work on other CGI based applications, including those running on Apple's OS X. :)

can insert this script.

```
#!/bin/bash
################ Begin Configuration ################
export LOGDIR=/tmp
################ End Configuration ################
INFILE=${LOGDIR}/in_${REMOTE_ADDR}
OUTFILE=${LOGDIR}/out_${REMOTE_ADDR}
set >> /tmp/debug
echo "********* Begin HTTP Dialog ************" >> $INFILE
echo "********* Begin HTTP Dialog ************" >> $OUTFILE
set >> $INFILE
echo "------------------" >> $INFILE
cat - | tee -a $INFILE | ./wtbbrk | tee -a $OUTFILE
echo "********* End HTTP Dialog ************" >> $INFILE
echo "********* End HTTP Dialog ************" >> $OUTFILE
```


The magic of it really lies in the tee program available on most UNIX operating systems (if not all of them.)

This program takes stdin data and makes a copy of it to some named file. It then sends the data along on stdout so that the next process in the pipe can receive it and do it's bit of work.

The work of this is done on the third from the bottom line in the script. We perform a “cat -” which means capture stdin and pass it along to stdout. That stdout is the stdin of the first tee which will take whatever is sent by the web browser and place it into a file.

The data is then passed along to the Progress wspd_cgi.sh script (mine is renamed wtbbrk and customized to look at a particular broker) which passes the

information to Webspeed.

Webspeed then manipulates that information based on your r-code and passes that information back out to the messenger who in turn passes it along via a pipe into the second tee command. That command will put the output in yet another separate file and then pass the results along to the web server which will in turn pass it along to the web browser.

The script places it's files by what you configure LOGDIR to be. The files will benamed in_[Your Remote Address] and out_[Your Remote Address]. The files are postfixed with the IP of the machine running the browser – this way multiple developers can figure out which in and out file is theirs[2].

Since under CGI, a lot of the information for an HTTP GET comes through the environmental variables, those are placed at the top of the HTTP message. *One should not see them as part of the HTTP message, but in what environment the message is being worked on under.*

This can also help you isolate the various components of an HTTP header that is coming in and out of the transaction server.

2With some fancy work on nslookup, you can transform the IP address into a work station name, provided they are assigned in DNS.

Example of using Cookies

Here is a simple program that will set a cookie:

```
<!--WSS

PROCEDURE OUTPUT-HEADERS:

  SET-COOKIE("ExampleCookie", "1234", ?, ?, ?, ?, ?).

END.

-->
<html>
<body>
Set the cookie!
<form method="post">
Label: <input type="text" name="MyFieldValue">
Label: <input type="text" name="MyOtherValue">
<input type="submit">
</form>
</body>
</html>
```

The input file reads as following. Note that since the request is a GET – there is not very much in the stdin portion of the data set – most of the information is stored in environmental variables.

```
********* Begin HTTP Dialog ************
BASH=/bin/bash
BASH_VERSION=1.14.7(1)
DOCUMENT_ROOT=/home/httpd/html
EUID=500
GATEWAY_INTERFACE=CGI/1.1
HOSTTYPE=i386
HTTP_ACCEPT=text/xml,application/xml,application/xhtml+xml,text/html;q=
0.9,text/plain;q=0.8,image/png,*/*;q=0.5
HTTP_ACCEPT_CHARSET=ISO-8859-1,utf-8;q=0.7,*;q=0.7
HTTP_ACCEPT_ENCODING=gzip,deflate
HTTP_ACCEPT_LANGUAGE=en-us,en;q=0.5
HTTP_CONNECTION=keep-alive
HTTP_HOST=amduus2
HTTP_KEEP_ALIVE=300
HTTP_REFERER=http://amduus2/online/logit/webtools/fileact.w
HTTP_USER_AGENT=Mozilla/5.0 (Macintosh; U; PPC Mac OS X Mach-O; en-US;
rv:1.7.3) Gecko/20040910
```

```
IFS=

INFILE=/tmp/in_172.16.1.34
LOGDIR=/tmp
OPTERR=1
OPTIND=1
OSTYPE=Linux
OUTFILE=/tmp/out_172.16.1.34
PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/X11R6/bin
PATH_INFO=/setcookie.html
PATH_TRANSLATED=/home/httpd/html/setcookie.html
PPID=19328
PS4=+
PWD=/home/appl/cgi
QUERY_STRING=
REMOTE_ADDR=172.16.1.34
REMOTE_PORT=57745
REQUEST_METHOD=GET
REQUEST_URI=/online/logit/setcookie.html
SCRIPT_FILENAME=/appl/cgi/logit
SCRIPT_NAME=/online/logit
SERVER_ADDR=172.16.1.36
SERVER_ADMIN=root@localhost
SERVER_NAME=Amduus2
SERVER_PORT=80
SERVER_PROTOCOL=HTTP/1.1
SERVER_SIGNATURE=<ADDRESS>Apache/1.3.12 Server at Amduus2 Port
80</ADDRESS>

SERVER_SOFTWARE=Apache/1.3.12 (Unix)  (Red Hat/Linux) PHP/4.3.1
mod_perl/1.21
SHELL=/bin/bash
SHLVL=1
TERM=dumb
UID=500
_=********* Begin HTTP Dialog ************
------------------
********* End HTTP Dialog ************
```

The more interesting information is in the out_* file:

********* Begin HTTP Dialog ************

```
Set-Cookie: ExampleCookie=1234; path=/online/logit
Content-Type: text/html

<!-- Generated by WebSpeed - http://www.webspeed.com/ -->

<html>
<body>
```

```
Set the cookie!
<form method="post">
Label: <input type="text" name="MyFieldValue">
Label: <input type="text" name="MyOtherValue">
<input type="submit">
</form>
</body>
</html>
********* End HTTP Dialog ************
```

In the above you can see the actual headers that Webspeed creates via the SET-COOKIE() function to tell the browser the name and value of the cookie in the HTTP header portion. Following that is the HTTP Message portion that contains the HTML that should be rendered on the screen.

Now lets use that page to post some information. Note that we will get the very same output file data because we aren't doing anything different.

BUT we are getting different data in the input file:

```
********* Begin HTTP Dialog ************
BASH=/bin/bash
BASH_VERSION=1.14.7(1)
CONTENT_LENGTH=62
CONTENT_TYPE=application/x-www-form-urlencoded
DOCUMENT_ROOT=/home/httpd/html
EUID=500
GATEWAY_INTERFACE=CGI/1.1
HOSTTYPE=i386
HTTP_ACCEPT=text/xml,application/xml,application/xhtml+xml,text/html;q=
0.9,text/plain;q=0.8,image/png,*/*;q=0.5
HTTP_ACCEPT_CHARSET=ISO-8859-1,utf-8;q=0.7,*;q=0.7
HTTP_ACCEPT_ENCODING=gzip,deflate
HTTP_ACCEPT_LANGUAGE=en-us,en;q=0.5
HTTP_CONNECTION=keep-alive
HTTP_COOKIE=ExampleCookie=1234
HTTP_HOST=amduus2
HTTP_KEEP_ALIVE=300
HTTP_REFERER=http://amduus2/online/logit/setcookie.html
HTTP_USER_AGENT=Mozilla/5.0 (Macintosh; U; PPC Mac OS X Mach-O; en-US;
rv:1.7.3) Gecko/20040910
IFS=

INFILE=/tmp/in_172.16.1.34
LOGDIR=/tmp
OPTERR=1
OPTIND=1
OSTYPE=Linux
OUTFILE=/tmp/out_172.16.1.34
```

```
PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/X11R6/bin
PATH_INFO=/setcookie.html
PATH_TRANSLATED=/home/httpd/html/setcookie.html
PPID=19326
PS4=+
PWD=/home/appl/cgi
QUERY_STRING=
REMOTE_ADDR=172.16.1.34
REMOTE_PORT=57767
REQUEST_METHOD=POST
REQUEST_URI=/online/logit/setcookie.html
SCRIPT_FILENAME=/appl/cgi/logit
SCRIPT_NAME=/online/logit
SERVER_ADDR=172.16.1.36
SERVER_ADMIN=root@localhost
SERVER_NAME=Amduus2
SERVER_PORT=80
SERVER_PROTOCOL=HTTP/1.1
SERVER_SIGNATURE=<ADDRESS>Apache/1.3.12 Server at Amduus2 Port
80</ADDRESS>

SERVER_SOFTWARE=Apache/1.3.12 (Unix)  (Red Hat/Linux) PHP/4.3.1
mod_perl/1.21
SHELL=/bin/bash
SHLVL=1
TERM=dumb
UID=500
_=********* Begin HTTP Dialog ************
------------------
MyFieldValue=This+is+input+1&MyOtherValue=This+is+input+two%21*********
End HTTP Dialog ************
```

Hopefully you can see way down at the bottom we got some information in a POST. That means the data from the form is sent through stdin to the CGI interface, where as a GET would see it in the QUERY environmental variable. Looks pretty similar to a GET doesn't it?

Example of catching an error that will cause the Internal Server Error page

Now lets get one of those bugger screens. An easy way to make this happen is to remove the `-weblogerror` in your srvrStartupParam entry of your ubroker.properties file.

Lets modify the program so that it will freak out the web server. We send an illegal argument to SET-COOKIE (the 1 in the fourth position) and a wrong data type to Login from OUTPUT-HEADERS.

```
<!--WSS

PROCEDURE OUTPUT-HEADERS:

  SET-COOKIE("ExampleCookie", "1234", ?, 1, ?, ?, ?).

  /* This will cause a run time error!  And that will produce */
  /* our Internal Server Error page to appear from the web    */
  /* server because we are sending a Progress error message   */
  /* in the HTTP header portion which will flip it out.       */

  RUN Login (INPUT "A String!").

END.

PROCEDURE Login:

  DEF INPUT PARAMETER iVarA AS INTEGER NO-UNDO.

END.

-->
<html>
<body>
Set the cookie!
<form method="post">
Label: <input type="text" name="MyFieldValue">
Label: <input type="text" name="MyOtherValue">
<input type="submit">
</form>
</body>
</html>
```

Why is it freaking out the web server? Take a look at the out_* file for your workstation and you can see this:

```
********* Begin HTTP Dialog ************
Received RECONNECT from WTB<P>
Set-Cookie: ExampleCookie=1234; path=/online/logit
Content-type: text/html
n** Invalid character in numeric input A. (76)<P>Incompatible datatypes
found during runtime conversion. (5729)
<P>Procedure OUTPUT-HEADERS setcookie.html sent sub-procedure Login
setcookie.html mismatched parameters. (2570)<P>
Content-Type: text/html
```

```
<!-- Generated by WebSpeed - http://www.webspeed.com/ --
><html><body>Set the cookie!<form method="post">Label: <input
type="text" name="MyFieldValue">Label: <input type="text"
name="MyOtherValue"><input type="submit"></form></body></html>
********* End HTTP Dialog ************
```

We can see from the bolded portions of the output file, that the HTTP header was populated with Webspeed errors! These items are illegal in the header portion and the web server knows it. So it throws out the dreaded Internal Server Error page.

Because you have this new tool, you can also see the errors as plain as can be!

Scott Auge is the founder of Amduus information Works. He has been working with Progress technologies since Version 6. He works with UNIX platforms dealing with integration and web based applications.

contact Scott Auge at sauge@amduus.com or see http://amduus.com/tresca

See http://amduus.com/serviceexpress/PricePerUser.html for pricing!

Publishing Information:

Scott Auge publishes this document. I can be reached at sauge@amduus.com.

Amduus Information Works, Inc. assists in the publication of this document by providing an internet connection and web site for redistribution:

Amduus Information Works, Inc.

1818 Briarwood

Flint, MI 48507

http://www.amduus.com

Other Progress Publications Available:

This document focuses on the programming of Progress applications. If you wish to read more business oriented articles about Progress, be sure to see the Profile's magazine put out by Progress software http://www.progress.com/profiles/

There are other documents/links available at http://www.peg.com .

There is a web ring of sites associated with Progress programming and consultants available at http://i.webring.com/hub?ring=prodev&id=38&hub .

White Star Software publishes a commercial document called "Progressions." It is simular to this document but with different content. More information can be found at http://wss.com/. White Star also publishes Progress Programming books!

Article Submission Information:

Please submit your article in OpenOffice[3] format or as text. Please include a little bit about yourself for the "About the Author" paragraph.

Looking for technical articles, *marketing Progress* articles, articles about books relevant to programming/software industry, white papers, etc.

Send your articles to sauge@amduus.com! Thanks!

3 OpenOffice is a freely available Office Suite for Windows, Apple, and *NIX based operating systems. You can download it at http://openoffice.org. This document is edited on OpenOffice.