

## Lesson 3 Outline - Script Troubleshooting

Almost anyone can install a well-documented CGI script. The hard part is figuring out what went wrong when it doesn't work!

- I. Types of failures
  - A. Server errors
    - 1. 500 – Internal Server Error
    - 2. 501 – Method Not Implemented
    - 3. 401 – File not Found
    - 4. 403 – Access Forbidden
  - B. Failure to execute script
    - 1. “No Data” message
    - 2. Script text is displayed instead of executed
    - 3. “Incomplete Script Headers” message
  - C. Unexpected behavior from script
    - 1. Mail isn't sent properly
    - 2. Script does not do what it is supposed to do
    - 3. Blank page is shown
    - 4. Files starts to download instead of showing the results page
  
- II. Sources of failures
  - A. Server configuration
    - 1. CGI not enabled
    - 2. Incorrect file extension or directory for CGI scripts
    - 3. Incorrect file or directory permissions
  - B. Syntax errors
    - 1. Un-escaped special characters (ex. ashley\@enscript.com)
    - 2. Missing semicolons, quotes, etc
    - 3. Control characters inside file (such as ^M inserted by binary upload of text files)
  - C. Incorrect configuration of script
    - 1. Wrong values for path/URL variables
    - 2. Incorrect path to system resources
    - 3. Incorrect #!/bin/Perl line
    - 4. Unable to find necessary Perl modules or required libraries
  - D. Programming errors
  
- III. Debugging methods
  - A. Command line / telnet
  - B. Print statements within script
  - C. Server logs
  - D. Debugging aids
    - 1. Small scripts to display environment, etc.
    - 2. Perl debugger
  
- IV. Defensive programming
  - A. Always check for return codes from function calls
  - B. Flush the output buffer at the beginning of the script
  - C. Print the Content-type header as early as possible
  - D. Use a debug flag to print environment information to the browser
  - E. Use “strict” in order to avoid common usage errors

Probably the most difficult skill is learning which debugging method to apply for various types of failures. Here is a reference chart that gives some general guidelines.

Symptom	Possible Causes	Debugging Tips
<b>“No Data” message</b>	<p>The script printed a Content-type header but crashed before returning any other information to the browser.</p> <p>A path through the script fails to return any output.</p>	<p>Insert many print statements throughout the main program to diagnose where the failure is occurring.</p> <p>Redirect STDERR to STDOUT temporarily.</p> <p>Make sure you have printed the Content-type header before you print any other information and that you return some type of result data from the script (HTML, binary data, plain text, etc.)</p>
<b>The text of the script is displayed instead of executed</b>	<p>The server is not executing the script but is interpreting it as a text file.</p>	<p>Check the server configuration requirements for CGI scripts. Often they must end in .cgi or be located in cgi-bin</p>
<b>A blank page is shown after the script is executed.</b>	<p>An incomplete HTML page was returned by the script; often this occurs if you have a complex table structure in the HTML. If the script has an error before finishing the HTML (ie. Before closing the table properly) then the entire page looks blank.</p>	<p>Try to “View Page Source” in the browser to see if you can tell where the HTML stopped.</p>
<b>A file starts to download instead of the results page being displayed.</b>	<p>The Content-type header is missing or is not followed by a blank line.</p> <p>You started printing to STDOUT before you printed the header.</p>	<p>Put another Content-type header print statement at the top of your script, then run the script again to see what is being printed.</p>
<b>Mail isn’t sent properly</b>	<p>The path to sendmail is not set properly.</p> <p>You are on an NT host and the script is written for UNIX .</p> <p>You do not have the appropriate mailer program installed.</p>	<p>Check the FAQ page for you ISP to see what mailer options are available.</p> <p>Try to find the location of sendmail (or whichever mailer you are trying to use) from the command line. On UNIX try using “which” or “find”.</p>
<b>“Incomplete Script Headers” message</b>	<p>Often shown on NT systems when there is a perl syntax error in the script.</p> <p>The Content-type header is missing or is not followed by a blank line.</p>	<p>Generally this error is accompanied by a more detailed error message, such as the Perl syntax error.</p>

<b>403 – Access Forbidden Error</b>	<p>The script is protected by htaccess and you did not enter the proper username or password.</p> <p>The script does not have read/execute permission.</p>	
<b>401 – File not Found</b>	<p>The script does not have read/execute permission.</p> <p>The script is not in the correct directory for CGI scripts.</p>	<p>Sometimes the cgi-bin directory is actually out side of the web root directory. Check with the ISP to see that you are uploading the CGI scripts to the correct place.</p>
<b>501 – Method Not Implemented</b>	<p>The script is not in the correct directory for CGI scripts.</p>	<p>Check the server configuration requirements for CGI scripts (ex. ScriptAlias path).</p>
<b>500 – Internal Server Error</b>	<p>The script does not have execute permission.</p> <p>Files accessed by the script do not exist or do not have the proper permissions.</p> <p>One of the perl built-in function calls failed (ex. file open).</p> <p>The script was uploaded in binary format.</p> <p>You are on an NT host and the script is written for UNIX.</p> <p>Incorrect path to system resources or Perl.</p>	<p>Check for Perl syntax errors from the command line.</p> <p>Try to run from the command line or from a “nobody” script to see if any run-time errors are returned by Perl.</p> <p>Check and print return values on function calls, especially open() calls.</p> <p>If you use the “   die” syntax in the script (and you’re not using the CGI::Carp module), try replacing it with “   print \$!” to see if you can capture any error messages.</p> <p>Redirect STDERR to STDOUT temporarily.</p>