International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Common Gateway Interface

Kamal Kathuria[1], Chaynika Kapoor[2], Apoorva Adlakha[3]

[1, 2, 3] Student, CSE, Dronacharya College of Engineering, Gurgaon, Haryana, India

Abstract: ***In the world of the Internet, Apache and Internet Information Server (IIS) were the web servers that were developed for exchanging the information between clients computer having different Operation System. The only function of IIS is that of displaying static information such as HTML and image files onto the Web Browser. But the major problem lies here that when the information is updated in the database, the administrator has to update it by manual operation. Because it is necessary to update several places about the same information, the work load becomes higher than it is assumed, thus the updation of error and omission may occur. Such problems faced by the Web Developers were solved by the use of a Common Gateway Interface (CGI) program such as a bulletin board system and a Blog system. However, these programs are opened to Internet and often they don't have user authentication and access control mechanism. This means that they have the problem that the user can access it easily and freely only by getting the URL and inputting it into a Web Browser. Common Gateway Interface (CGI) is a standard method used to generate dynamic content for the Web pages and Web applications. When implementation of CGI is done on a Web server, it provides with an interface between the Web server and programs that generate the Web content. These programs are popularly known as CGI scripts or simply CGIs; they are usually written in a scripting language, but can also be written in any other programming language.***

Keywords: Common Gateway Interface(CGI), packet filtering, DACS Scheme, Websites, PBNM, Destination NAT

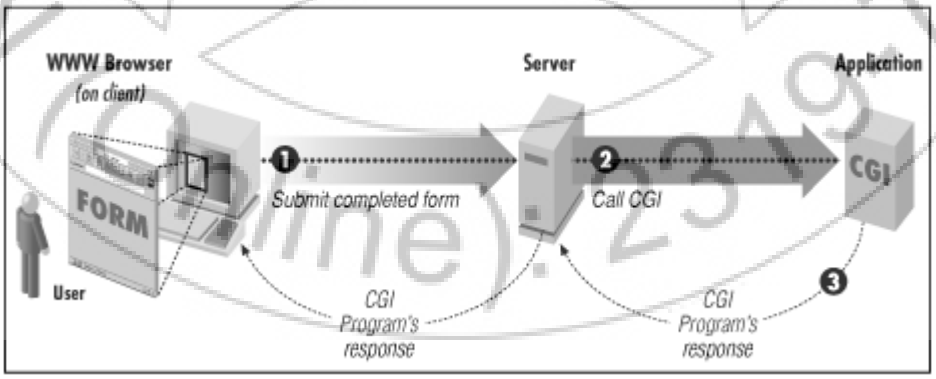
1. Introduction

As we traverse the vast frontier of Internet commonly known as the "World Wide Web", we generally come across many documents that surprise us, *"Can this thing really happen?"* These documents can be like forms that ask us for the feedback of any particular thing or registration information just like a image maps, registration forms that allow us to click on different parts of the image, counters for displaying the number of views to the document, and utilities that allow us to search databases for particular information. In many cases especially on an animated or good looking websites, we find that these effects were achieved using the "Common Gateway Interface", which is commonly known as "CGI".

One of the Internet's worst-kept secrets is that CGI is astonishingly simple. That is, its design is trivial, and anyone with basics or we can say anyone with an iota of programming experience can pen down the rudimentary scripts that work. It is used only when a person is short of time and his needs are more demanding ,in that case you have to master the more complex workings of the Web. In a way, we can say that CGI is as easy as cooking is: anyone can bake a muffin or to boil an egg. It's only when you want a mayonnaise sauce that things start to get complicated.

CGI is the part of the Web server that can communicate with other programs running on the server. With the help of CGI, the Web server can call up a existing or a running program, while passing user-specific data to the program. The called program then processes that data and the server passes the program's response back to the Web browser.

CGI isn't magic; it's just programming with some special types of input and a few strict rules for program output. This is a combination of programming and some special skills that one possesses. Underlying it all is the simple model shown in



2. Implementing CGI Scripts

The following CGI script is created using Perl. This script tabulates to create the table of output
following the script:

```
#!/usr/local/bin/perl
print "Content-type:text/html\n\n";
open(INF,"votes.out");
@NAMES = <INF>;
close(INF);
foreach $line (@NAMES)
{
 $linecount++;
```

```
 @values = split(/\|/, $line);
 foreach $value (@values)
 {
 $FORM{$value}++;
 }
}
print "<html><head><title>Current Results</title></head>\n";
print "<BODY BGCOLOR=\"#AABBBB\" TEXT=BLACK LINK=\"#001170\"
VLINK=\"#001170\" ALINK=\"#001170\">\n";
print "<h2>Current Results</h2><BR><BR>\n";
print "<TABLE WIDTH=400 BORDER=1 CELLSPACING=0 CELLPADDING=0
BGCOLOR=\"#779E9E\">";
print "<TR><TD colspan=1 align=center><h3>Boy Names:</h3></TD>";
print "<TD colspan=1 align=center><h3>Girl Names:</h3></TD>";
print "<TR BGCOLOR=\"#88AFAF\"><TD><TABLE WIDTH=200 BORDER=0
CELLSPACING=0 CELLPADDING=0";
print " BGCOLOR=\"#779E9E\">\n";
@boynames = ("boyname q","boyname r","boyname s");
foreach $x (@boynames) {
 {
 print "<TR BGCOLOR=\"#88AFAF\"><TD WIDTH=33%> </TD>";
 print "<TD WIDTH=30%>$x</TD><TD WIDTH=4%><B>$FORM{$x} </B></TD>";
 print "<TD WIDTH=33%> </TD></TR>\n";
 }
}
print "</TABLE></TD><TD>";
print "<TABLE WIDTH=200 BORDER=0 CELLSPACING=0 CELLPADDING=0";
print " BGCOLOR=\"#779E9E\">\n";
@girlnames = ("girlname q", "girlname r", "girlname s");
foreach $x (@girlnames) {
 {
 print "<TR BGCOLOR=\"#88AFAF\"><TD WIDTH=33%> </TD>";
 print "<TD WIDTH=30%>$x</TD><TD WIDTH=4%><B>$FORM{$x} </B></TD>";
 print "<TD WIDTH=33%> </TD></TR>\n";
 }
}
print "</TABLE></TD></TR>";
print "<TR><TD COLSPAN=2 BGCOLOR=\"#779E9E\" ALIGN=RIGHT><B>$linecount</B> people had voted.</TD></TR>";
print "</TABLE>";
print "<BR><BR><BR>\n";
print "<A HREF=\"http://www.beth.cx/baby/\">Go back to Beth v2.0</A>\n";
print "</DIV>\n";
print "</BODY></HTML>\n";
```

3. Why CGI?

The above applications can be implemented using other languages as well for e.g., server-side JavaScript .ACGI, DHTML , VRML, PHP, but many of these given languages are developed after CGI. CGI, then, has become a standard, and many programmers prefer simply to “tweak” their old CGI scripts for new purposes, in spite of starting from the scratch with the new languages. Also, CGI is more versatile than other languages in many ways. A traditional CGI application made using Perl language, for instance, can run on a large number of platforms with wide variety of Web servers.

CGI also has disadvantages . Many of the new languages developed in response to CGI being slow, so these run at a faster rate more efficiently. Also, CGI is not secured ,it has many security issues. Since a file that uses CGI is executable, it is equivalent to letting anyone in the world run a program on your machine.

Obviously, this is not the safest thing to do. For various security reasons, many Web hosts do not allow users to run these CGI scripts. In this case, though, you can have your CGI applications to be hosted for you remotely. Http://www.hypermart.net is a free host which allows CGI scripting, and http://cgi.resourceindex.com/Remotely_Hosted/ lists a number of other hosts that allows CGI.

Related is the fact that the programs that uses CGI scripts need to reside in a exceptional directory, so that the server knows basically which program to execute rather than simply display it to the browser. This directory, commonly /cgi-bin, is under the direct control of the Webmaster. This prohibits the average user from creating and running programs that use CGI.

CGI programs are mostly used with HTML Languages and its FORM's, and it provides the server interface that receives the form variables and processes them for the users. The requirement for being able to act as a CGI program is the ability to read from Standard Input (stdin), or the ability to access Environment variables.

Access to CGI programs and scripts must be made via the "cgiwrap" process. The below steps describe how to implement a program using "cgiwrap".

To use the "cgiwrap" process CGI programs are mostly used with HTML Languages and its FORM's, and it provides the server interface that receives the form variables and processes them for the users. The requirement for being able to act as a CGI program is the ability to read from Standard Input (stdin), or the ability to access Environment variables.

Access to CGI programs and scripts must be made via the "cgiwrap" process. The below steps describe how to implement a program using "cgiwrap".

To use the "cgiwrap" process:

1. Create a sub-directory within your "public_html" directory called "cgi-bin".
2. The directory should have permissions of 0711. Moreover It should be world executable rather than world readable.

3. Place your executable scripts and/or programs within this directory. These files must have permissions 0700. They must be only executable by you.
4. Code the "ACTION" URL within your "<FORM>" tag as:
5. <FORM ACTION="/cgi-bin/cgiwrap/username/CGIprogram"
   METHOD= [GET | POST] >
   "username" is your user ID that you use to login. "CGIprogram" is the name of your executable script or program. Note that the username in the "<FORM>" tag does not start with a tilde (~) character unlike that in the case of URL for your Home Page.
   - " METHOD=POST" can only be used within a "<FORM>" tag.
   - "METHOD=GET" can be used directly within a "<FORM>" tag and implicitly in the URL of an HTTP link.
   - " METHOD=POST " provides any parameters as "keyword=value" pairs in a single input line in Standard Input (stdin).
   - " METHOD=GET" provides the same input string in an Environment variable called "QUERY_STRING".
   - Some browsers also provide a value for an Environment variable called "CONTENT_LENGTH" This should be treated as informational only as this is not universal
   - Multiple keyword=value pairs are separated by a single ampersand (&) character.
   - Blanks or spaces in keywords or values are received as a plus (+) sign.
   - Most special characters are received as a two-digit hexadecimal value preceded by a per-cent (%) sign.
   - Multi-line values have their lines separated by a carriage-return/line-feedpair, encoded in hexadecimal as "%0D%0A".
6. Note that the URL in this tag does NOT specify the name of a Web server.
7. The "cgiwrap" process does a few basic security checks so that it can be prevented from the world and then executes our script or program, running under our user ID. The program or script must:
   - be executable;
   - not be setuid or setgid;
   - be a physical file in "~username/public_html/cgi-bin";
   - not be a symbolic link to any other file.
   - be owned by the username listed in the "ACTION" URL.
8. Since the "cgiwrap" process executes your script or program under your personal or own user ID, the program or script have the same access to files as you do when you are logged on.
9. This means that any files that you need to access should be only writable by you. They should not be world writable!
   You can create sub-directories within your cgi-bin directory. These should also have permission 0711. In this case, code the "ACTION" URL within your "<FORM>" tag as:
   <FORM ACTION="/cgi-bin/cgiwrap/username/Directory/CGIprogram">
10. If you need full control of the HTML Headers produced by your script/program, you can substitute "nph-cgiwrap" for the normal cgiwrap program. This is mostly needed only if you wish to create your own error handlers for the problems with your owhn personal web content. Example- If you want to provide your own handling for a "404 - page not found" error. Note that this support will require other things. If anyone wants to do this then e should read the Apache Manual very closely .Hint: The Webmaster will not help you!

References

[1] http://adashimar.hypermart.net/ -- introduction to CGI; focuses on steps for installing
[2] http://bignosebird.com/cgi.shtml -- free Perl CGI scripts
[3] http://cgi.resourceindex.com -- CGI-related resources, including scripts
[4] (/Programs_and_Scripts/), documentation (/Documentation), and information on remote
[5] hosting for CGI (/Remotely_Hosted/)
[6] http://developer.netscape.com/viewsource/lazar_cgi.html -- CGI vs. server-side
[7] JavaScript for databases
[8] http://hoohoo.ncsa.uiuc.edu/cgi/overview.html -- CGI documentation (including a good
[9] intro page (/intro.html) and examples
[10] http://hotwired.lycos.com/webmonkey/99/26/index4a_page4.html?tw=programming --
[11] CGI permission levels
[12] http://jgo.local.net/LinuxGuide/linux-chmod.html -- CGI permission levels and chmod
[13] http://worldwidemart.com/scripts/ -- free CGI scripts
[14] http://www.cpan.org/ -- Comprehensive Perl Archive Network; documentation, FAQ,
[15] mailing list, scripts
[16] http://www.hypermart.net -- Web host that allows CGI scripting http://www.linkyours.com/cgi_overview.html -- graphic overview of CGI interface
[17] process
[18] http://www.mattkruse.com/info/cgi/ -- good, simple introduction to CGI
[19] http://www.w3.org/Security/Faq/wwwsf4.html -- World Wide Web Security FAQ on
[20] CGI scripting