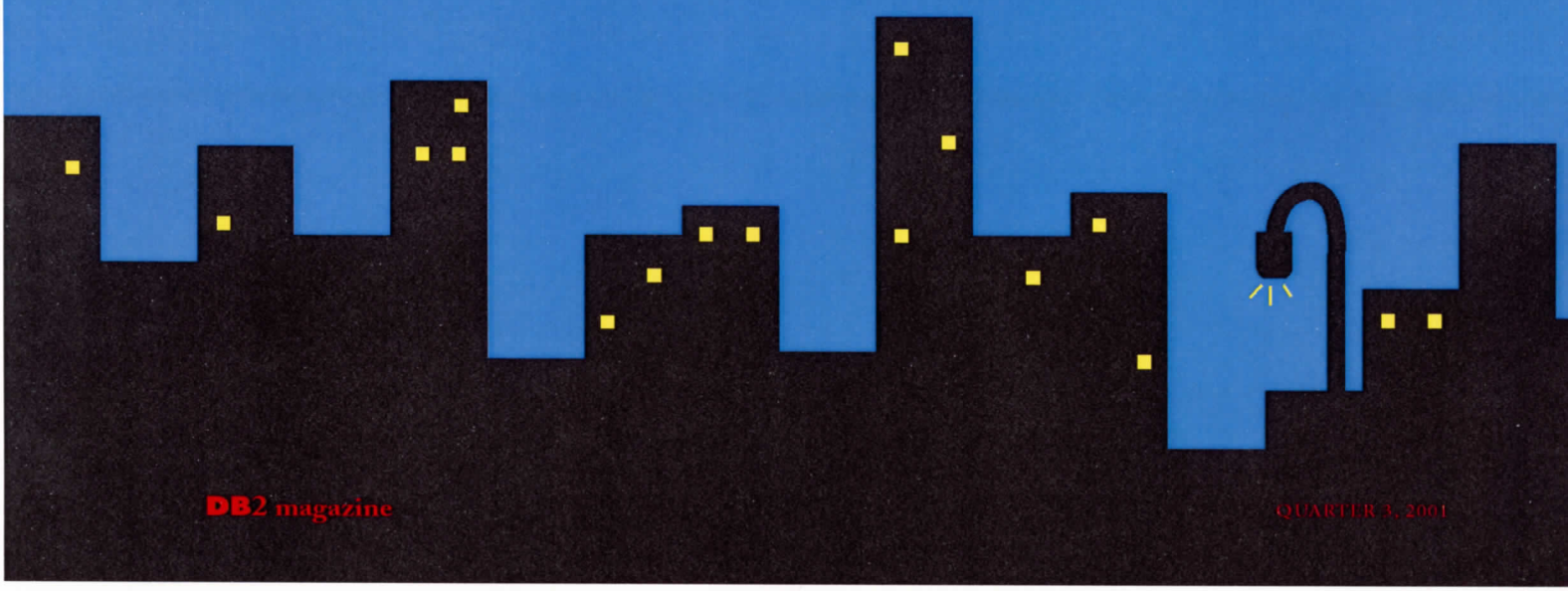
Dynamic Duo

GLEN JOHNSON

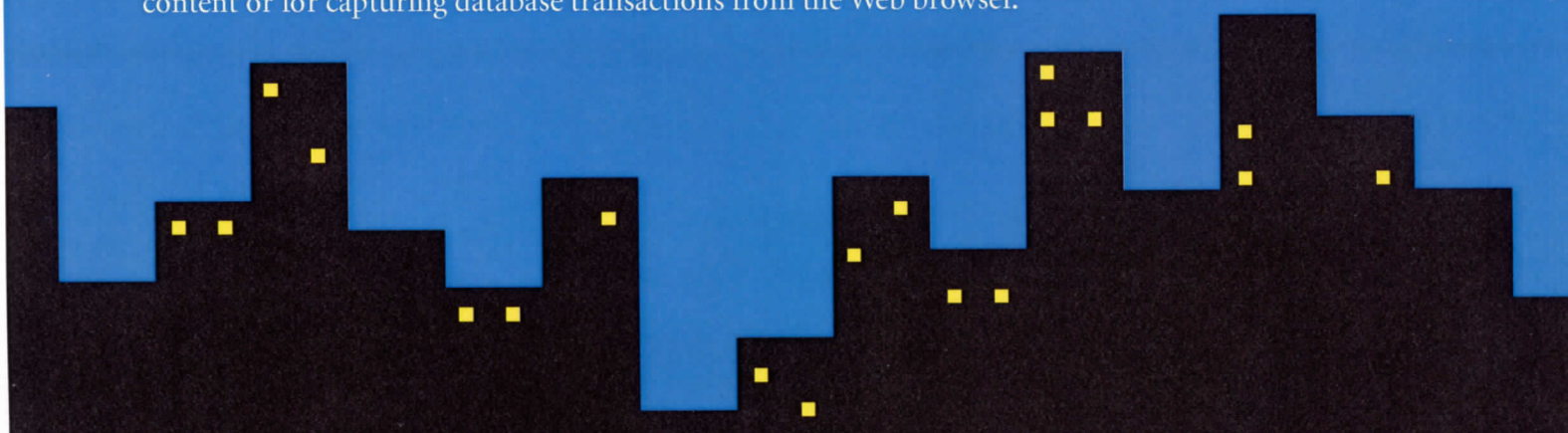
PHP and DB2 for Linux enliven Web sites with interactive possibilities. Here's how to put them to work.



PHP DB2

Without a doubt, Web-enabled applications have been a driving force in the phenomenal growth of the Internet. The success of the server components of those applications owes much to the underlying architecture provided by the open source operating system GNU/Linux (Linux) and the Apache Web server project, a collaborative software development effort that created a commercial grade, freely available source code implementation of an HTTP Web server.

As Web-based applications matured, it became clear that Web servers that could only render static HTML could not meet application needs. Most Web applications today need to incorporate application logic and access data stored in relational databases. This data can be used for the generation of customized dynamic content or for capturing database transactions from the Web browser.



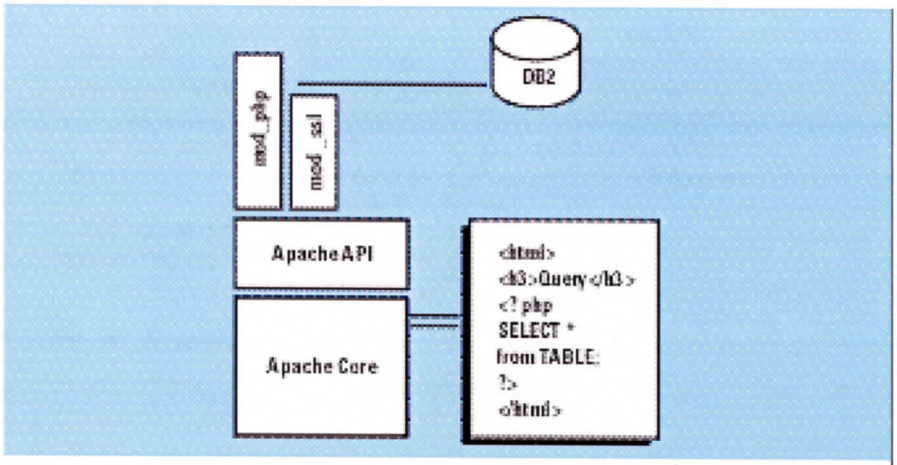


Figure 1: An Apache/PHP/DB2 environment.

```
##   pre-configure Apache for PHP4's configure step
     cd apache_1.3.19
     ./configure \
          --prefix=/usr/local/apache
##   build PHP
     cd ../php-4.0.4pl1
      ./configure \
          --with-apache=../apache_1.3.19 \
          --with-mysql \
          --with-ibm-db2=/usr/IBMdb2/V7.1 \
          --with-xml \
          --enable-debug=yes
     make
     make install
     cd ..
##   build openssl
     cd openssl-0.9.6
     ./config
     make
     cd ..
##   build modssl
     cd  mod_ssl-2.8.1-1.3.19
     CFLAGS='-O2 -I/apachessl/openssl-0.9.6/include  '\
     ./configure \
     --with-apache=../apache_1.3.19 \
     --with-ssl=../openssl-0.9.6 \
     --activate-module=src/modules/php4/libphp4.a \
     --enable-module=php4 \
     --prefix=/usr/local/apache
     cd ..
##  Finally build Apache
    cd apache_1.3.19
    make
##  make a test ssl certificate and create pass phrase for ssl
server
    make certificate TYPE=custom
##  and put apache in /usr/local/apache
    make install
```

Listing 1: Using gcc to build the Apache server.

Apache's design allows it to grow to meet these demands through such modules as the PHP: Hypertext Processor (PHP) language, a server-side,cross-platform,HTML embedded scripting language for creating dynamic Web content.

In this article, I'll describe how to build an Apache Web server with PHP and DB2 connectivity in a Linux environment.I'll show you how to set up DB2 and obtain source code for Apache, PHP, and Secure Sockets Layer (SSL), and how to compile with DB2 support. At the end of the article, you will have created an environment similar to the one shown in Figure 1. First, though,I'll provide a bit of history on the influential Apache and Linux projects.

WHAT'S IN A NAME

Apache derives its name from the patches (extensions and bug fixes) that Web developers applied to the HTTP daemon (HTTPd) domain server developed at the National Center for Supercomputing Applications at the University of Illinois, Urbana-Champaign — then the most popular Web server software. A group of webmasters got together in 1995 to share and coordinate their changes and formed the original Apache (for a "patchy" server) Group. (You can read more about the Apache project at www.apache.org.)

The collaborative nature of the Internet fosters this sort of development process. Open source software such as Apache benefits from the patches provided by developers who use and support it. Today, Apache garners an amazing 60 percent of the Web server software market share, according to the Netcraft Web Server Survey (www.netcraft.com/survey). Similarly, Linux is the now the fastest growing server operating system.

The GNU (which stands for GNU's Not Unix) Project,started in 1984, arose from an effort to develop a free, Unix-like operating system. The best-known GNU variants use the kernel Linux and tend to be known by the name Linux,although GNU/Linux is more correct.

WHAT IS PHP?

Originally conceived by Rasmus Lerdorf, the open source PHP module, written in C, can be compiled and linked dynamically or statically with Apache. PHP syntax, patterned after Perl and C, allows application logic to be embedded directly into an HTML file. The PHP Apache module runs as part of the Apache (HTTPd) process, resulting in improved performance over CGI programs that run in a separately spawned process.

Although PHP is most commonly integrated with the Apache Web server in a Linux environment, it works with other Web servers (including THTTPd and AOLserver). It also works in other Unix environments and on Microsoft Windows. One of PHP's most useful features is its ability to connect natively to

many popular relational database engines,including DB2 Universal Database (UDB), Informix, Oracle, MySQL, and PostGresSQL.

DB2 for Linux combined with PHP and Apache provides a best-of-breed solution for building high-performance Web-enabled database applications. PHP code embedded in an ordinary static HTML Web page enables dynamic content generation by connecting with a DB2 data store. Users can receive customized Web content or take advantage of any DB2 feature via the Web. For example, if you have a workorder system using DB2 as a data store, you can use PHP to let customers enter transactions into this system over the Web. Or, you can use customer data profiled in DB2 to tailor dynamic Web content to those customers.

BASIC DATABASE SETUP

To create the environment Figure 1 shows, you'll first need to install DB2. You can find installation information in the article "Linux and DB2 UDB: Making the Match Work," by Paul Zikopoulos (*DB2 Magazine*, Quarter 4, 1999: Winter, available online at www.db2mag.com), and in the DB2 Administration Guide.

After installing DB2, create an instance in which to run your DB2 database. Still as root, type:

```
cd to /usr/IBMdb2/V7.1/instance and
  run db2icrt -u db2adm7 db2adm7
```

(Note: `db2adm7` is repeated twice. For the sake of simplicity, the DB2 instance owner and instance name are the same in this example.)

If the instance is created you will see this message:

```
DBI10701  Program db2icrt completed
  successfully
```

COMMAND LINE PROCESSOR (CLP)

To test your installation, open up an xterm or other terminal window and, if needed, enter `su - db2adm7` to log in as the DB2 instance owner and set up the correct environment. The db2install script should have added the following lines to your .profile:

```
# The following three lines have
  been added by UDB DB2.
if [ -f sqllib/db2profile ]; then
    ./sqllib/db2profile
fi
```

Keep in mind that all processes spawned by the Apache Web server need to inherit this DB2 environment; therefore, you should also add those three lines to root's .profile. They will be needed later.

You can now start DB2 with:

```
db2start
```

And DB2 will respond with:

```
SQL8007W  There are "90" day(s) left
in the evaluation period for the
product "DB2 Enterprise Edition".
For evaluation license terms and
conditions, refer to the IBM
Evaluation Agreement in the
EVALUATE.AGR file, located in the
following directory:
"/usr/IBMdb2/V7.1/Readme/en_US".
SQL1063N  DB2START processing was
successful.
```

If you are entitled to use another license

```
<?php
// quick test to see if Apache/PHP/DB2 works ...

if ($action=="")
{
echo "<html>";
echo "<h3>Test Connection to DB2 and Submit a Query </h3>";
echo "<pre>";
echo "<form action =$PHP_SELF method=post>";
echo "Database Name: <input type=\"text\" name=\"dsn\"
value=\"sample\"> <br>";
echo "         User: <input type=\"text\" name=\"user\"
value=\"db2adm7\"> <br>";
echo "     Password: <input type=\"password\" name=\"pass\"
><br>";
echo "       Select: <input type=\"text\" name=\"do_what\"
value=\"Select * from employee\" size=65  ><br>";

echo "         <input type=\"submit\" name=\"action\"
value=\"query\">";
echo "</form>";
}

if ($action=='query'){
echo "odbc_pconnect(".$dsn." ,".$user.",".$pass.")";
$id=odbc_pconnect($dsn,$user,$pass);
echo "\n ODBC connection id: ".$id;
$result=odbc_exec ($id,$do_what);
odbc_result_all($result);
odbc_close($id);
echo "<br>";
}

echo "</html>";
?>
```

Listing 2: Testing the Apache/PHP/DB2 combination.

for DB2 you may enable it by entering:

```
/usr/IBM/db2/7.1/adm/db2licm
  'LicenseFile'
```

Or by adding the license nodelock file to:

```
/var/lum/nodelock
```

To determine if everything is working properly, install the DB2_sample database with the command:

```
db2sampl
```

This script does not return a message when it successfully creates the DB2 sample database. However, once the sample database is created, you can interact with it using the CLP. Enter:

```
db2 "connect to sample"
```

and DB2 will respond with:

```
Database Connection Information

 Database server    = DB2/LINUX 7.1.0
 SQL authorization ID   = DB2ADM7
 Local database alias   = SAMPLE
```

The references at the end of this article provide more details and variations on installing DB2

COMPILING APACHE

As the root user, download the source code for the Apache Web server, SSL, and PHP and extract it into a convenient directory where it can be compiled (see the resources at the end of this article for places to download the necessary code).

Make sure your Linux installation includes a C compiler (such as gcc, available from from www.gnu.org). I typically create a directory called `apachessl` and use `wget` to download the source code (`wget` shows a nice progress indicator as it performs the download):

```
  cd apachessl
     wget httpd.apache.org/dist/
httpd/apache_1.3.19.tar.gz
     wget ftp.modssl.org/source/
mod_ssl-2.8.1-1.3.19.tar.gz
     wget  www.openssl.org/source/
openssl-0.9.6.tar.gz
     wget  us.php.net/
distributions/php-4.0.4pl1.tar.gz
     gzip -d -c apache_1.3.19.tar.
gz | tar xvf -
     gzip -d -c mod_ssl-2.8.1-1.3.
19.tar.gz | tar xvf -
     gzip -d -c openssl-0.9.6.tar.
gz | tar xvf -
     gzip -d -c php-4.0.4pl1.tar.
gz | tar xvf -
```

You should now have four source directories (apache_1.3.19, php04.0.4pl1, openssl-0.9.6, and mod_ssl-2.8.1-1.3.19). If you choose a later version of the code, the directory names will change slightly. Perform the exact steps in Listing 1 in each directory to use the gcc C compiler to build your Apache Server. (The lines beginning with ## are comments.)

You need to make a few minor changes to the stock Apache configuration file to accept the PHP module.

In `/usr/local/apache/conf/httpd.conf`, find the line that says:

```
# And for PHP 4.x, use:
#
```

Make sure the two `AddType` directives that follow are uncommented, as in this example:

```
   AddType application/x-httpd-php
.php
   AddType application/x-httpd-php-
source .phps
```

If you are already running a Web server on port 80, you will need to change the default configuration to use another port or stop running your current server on port 80. For example, the following stanzas of `/usr/local/apache/conf/httpd.conf` show how to configure the server to use port 81 for HTTP and port 443 for HTTPS:

```
#
# Port: The port to which the
standalone server listens. For
# ports < 1023, you will need httpd
to be run as root initially.
#
Port 81

##
```

```
                    $Fields = odbc_num_fields($result);
            print "<table border='1' width='60%' ><tr>";
            // Build Column Headers from DB2 table field names
            for ($i=1; $i <= $Fields; $i++){
            printf("<th bgcolor='grey'>%s </th> ",
    odbc_field_name( $result,$i));
            }
            print("</tr>");
            // Table Body
            while( odbc_fetch_row( $result )){
             print "<tr>";
             for($i=1; $i <= $Fields; $i++){
             // Data from DB2
             printf("<td>%s</td>\n",odbc_result( $result, $i ));
             }
             print "</tr>";
            }
            print "</table>";
```

Listing 3: Increasing HTML sophistication with PHP.

```
##  SSL Support
##
##  When we also provide SSL we have
to listen to the standard HTTP
##  port (see above) and to the
HTTPS port
##
<IfDefine SSL>
Listen 81
Listen 443
</IfDefine>
```

The Apache server will be running as user nobody, so make sure nobody has r-x privileges on the DB2 executables under /home/db2adm7/sqllib.

STARTING THE APACHE SERVER

Next start up the Apache server as root with:

```
/usr/local/apache/bin/apachectl
  startssl
```

You will see:

```
Apache/1.3.19 mod_ssl/2.8.1 (Pass
Phrase Dialog)
Some of your private key files are
encrypted for security reasons.
In order to read them you have to
provide us with the pass phrases.

Server yourserver.yourdomain.com:443
(RSA)
Enter pass phrase:
```

Just enter the pass phrase that you entered during the "make certificate" part of the build. The server should respond with:

```
Ok: Pass Phrase Dialog successful.

/usr/local/apache/bin/apachectl
startssl: httpd started
```

You can also start up the Web server in non-SSL mode without a pass phrase, with:

```
/usr/local/apache/bin/apachectl
  start
```

As a simple test, create a file called test.php with the following contents and place it in the /usr/local/httpd/htdocs directory (note the tags that distinguish the PHP code from HTML):

```
<html>
<?php
phpinfo()
?>
<html>
```

Point your favorite Web browser at the server:

```
https://yourserver/test.php
```

or

```
http://yourserver/test.php
```

You should see details about the PHP module running on your Apache server.

To actually test your connection to the DB2 sample database and retrieve data, put the code in Listing 2, which shows a list of names from the employee table of the sample database, into a file called testDB2.php and load it into your browser.

You can type other queries into the select box, too. This script barely scratches the surface of what can be done with PHP. For the next level of sophistication, peruse the documentation for the PHP functions at www.php.net/manual/en. For example, to obtain a finer degree of control over the HTML that PHP outputs, replace the line containing odbc_result_all($result) from Listing 2 with the code in Listing 3.

Doing so results in an HTML table generated on the fly and dynamically populated with data from the DB2 query. You can add a style sheet or otherwise tweak the HTML output to achieve whatever appearance you desire.

After you've mastered performing queries and processing their output,the next step is to try INSERT or UPDATE into the DB2 database tables. To do so, simply change the SQL statement in the $do_what variable on the example in Listing 3.

CONNECTING TO DIFFERENT SERVERS

You can use your Apache/PHP server to connect to a DB2 database on different server. Let's say your DB2 sample database is on machine www1, but your Apache/PHP server is on www2 (a DB2 client). To get your DB2 client to communicate with DB2 on the www1 server, you'll need to run this line on www1:

```
db2 "get database manager
  configuration" | grep SVCENAME
```

It should return something like:

```
TCP/IP Service name  (SVCENAME) =
  50000
Note the port number if it is not
  50000.
```

If SVCENAME is not defined, you need to define it to DB2 with:

```
db2 "UPDATE DATABASE MANAGER
  CONFIGURATION USING SVCENAME
    50000"
```

Make sure DB2COMM=tcpip is part of the DB2 environment when you run db2start.

Now, on www2 (the DB2/Apache/PHP client), run:

```
db2 "catalog tcpip node www1 remote
  www1 server 50000"
db2 "catalog database [server
  database name ] as [client
```

```
database name] at node www1"
```

The server database name could be `sample` if you are still using the sample database; the client database name could be `rsample`. You should now be able to connect to the remote database on www1 from www2 with:

```
db2 "connect to [client database
  name] user [userid] using
    [passwd]"
db2 "list tables"
```

TROUBLESHOOTING

Most often, trouble working with DB2/Apache/PHP results from an incorrectly set up DB2 environment. A typical symptom is PHP rendering a line of nonsensical characters to your browser. To avoid this common error, make sure that you run `/home/db2adm7/sqllib/ db2profile`. The root user must inherit this environment when starting Apache. Apache will then hand over control to the `nobody/nogroup` user who *only* has authority to access the DB2 database when the correct userid and password are passed from the PHP application.

You may also run into an error message such as:

```
SQL10007N Message "-1390" could not
  be retrieved.  Reason code: "1".
```

This message is most likely returned because the `DB2INSTANCE` environment variable is not set. Running the `db2profile` script will fix this. In our example, `DB2INSTANCE=db2adm7`.

Once `DB2INSTANCE` is set, you can issue `db2 "?SQL10007N"` to get more information about this or other error messages for which DB2 may initially only display error codes. Simply pass in the error number whose message you need to retrieve.

BEYOND STATIC

The Apache Web server and its associated Web-based applications have gone a great distance in expanding the usefulness of the Internet. The PHP module for Apache dramatically extends the reach of such applications by allowing them to interact with relational databases such as DB2 UDB, which in turn can provide tailored dynamic content or access to transactional data. Because PHP's syntax is patterned after Perl and C, programmers and nonprogrammers alike use it to great effect with a very shallow learning curve. ●

Glen Johnson is a software engineer with IBM's Linux Technology Center in Austin, Texas. He is responsible for the development of Web-enabled database tools for workflow automation and business process improvement. Prior to joining IBM in 1992 he was CFO of Nova Concepts Inc. and head of Immunochemsitry Research at Kallestad Laboratories. You can reach him at gjlynx@us.ibm.com.

RESOURCES:

PHP source code
www.php.net
PHP ODBC functions
www.php.net/manual/en/ref.odbc.php
PHP background
www.zend.com/zend/hof/rasmus.php
Sample PHP code
www.phpbuilder.com
DB2 UDB for Linux
ibm.com/software/data/db2/linux
www.linuxdoc.org/HOWTO/DB2-HOWTO
Apache source code
www.apache.org
SSL
www.modssl.org
www.openssl.org
GNU/Linux kernel
www.kernel.org