VBStats V3.1

Windows Web Server Statistics Reporter

By: Bob Denny <rdenny@netcom.com>
December 13, 1994

This document could have been three times this size. My apologies to those of you who are brand new to being a Web server operator, and have difficulties with the complexity of this. I just don't have the time to do any more than I already have.

Note to Beta Testers:

The database for 3.1 contains changes to almost all of the QueryDefs. This means that your old databases are not compatible with the 3.1 executables. My deepest apologies for this. I made the changes in order to achieve a substantial speed increase in most operations on large databases (over 20,000 access records).

If you saved your old NCSA/CERN flat-file logs, you can re-import them into a new database. If not, and you have Access 2.0, create a new empty database with the Setup Wizard and then use Access to import *all* of the QueryDefs from the new database into your existing live database. If you don't have Access 2.0, you'll have to use Visual Data do do the same thing. Save a copy of your old database! Create a new empty database with SetupWizard, then use Visual Data to delete all of the QueryDefs from your live database. Then, one by one, copy the SQL from the QueryDefs in the new database to QueryDefs in your live database by the same name. This will take a long time, I'm sorry. I just didn't have the time to write a conversion utility.

Introduction:

This kit contains five programs that together provide the ability to analyze Web server logfiles (in the NCSA/CERN "Common Log Format" *only*), and produce HTML pages (and a usage graph) describing the server usage. The basic idea came from the "wusage" program developed by Tom Boutell, however this program works entirely differently. The flat-file log information is brought into an Access 2.0 database, then a SQL-driven report generator makes the HTML pages and the graph.

You do not need to own Visual Basic or Access to use this package.

The kit is designed to optionally work automatically with Windows httpd 1.3 or later, checking to see if the server is running and if so, sending the "cycle logfiles" signal to the server, then sucking the cycled-out log data into the database. Of course, the kit can be used to analyze and report on data from other kinds of servers (NCSA, CERN, etc.).

There are five main programs:

Setup Wizard (SETUPWIZ.EXE):

Use this to do the initial configuration and customization of the package. You can also create an initial database with this program.

Log Converter (LOGTODB.EXE):

Captures common log format files and loads the database (creating it if needed). Run this periodically to transfer the flat-file logs into the Access database. Optionally translates IP addresses to hostnames

if you care, and if you have this disabled in the server (default, recommended).

Restriction Editor (RESTRICT.EXE):
A GUI "object restriction" editor. Use this to control the effect of classes of URL objects, hosts, and authenticated users have on the statistics. "Restriction" is the process of hiding objects (users, URL targets, and sites) from consideration in statistics summaries. Objects can be hidden from top-10 lists (but still count in the totals) or completely hidden (not even count in the totals).

Report Generator (REPORTER.EXE):
The program that generates the weekly HTML reports and a graph of weekly access counts from the raw data in the database.

Maintenance Utility (DBMAINT.EXE):
This program is used to clear old data from the database, to compact the database, and to clear the PastTotals table so all old reports will be regenerated the next time Reporter is run.

Supporting Programs:

In addition to the main programs, the kit includes the following supporting programs:

Windows Cron (WINCRON.EXE):
A program that can be used to schedule daily transfers of data from the server's flat-file logs into the database, and to do the weekly report/graph generation. Other Windows-based schedulers can be used as well.

Visual Data (VISDATA.EXE):
The data access sample program from Visual Basic, modified to work with Access 2.0 databases, and to permit inspection of "QueryDefs" in the database by clicking on the QueryDef in the Table/Query listbox. Use this for ad-hoc querying of the database if you don't own Access 2.0.

Sources: The sources for the main programs are available at the Windows httpd home FTP site at:

```
ftp://ftp.alisa.com/pub/win-httpd/util-support/vbs31src.zip
```

In order to work with the sources, you will need Visual Basic Professional V3.0. **You do not need to have Access 2.0 to work with the sources or the database.**

Installation:

You probably already did this, but just in case... Simply run the SETUP.EXE program and follow the directions. *No changes are made to your autoexec.bat or config.sys.*

IMPORTANT NOTE

The installation installs the files needed to use Microsoft Access 2.0 relational databases from Visual Basic into your "system" directory. If you already have VB Professional or Access 2.0, this installer will not replace newer versions of the DLLs. However, if you have Access 2.0, and yet you are using Visual Basic 3.0 data access features *without* the "compatibility layer" for VB, this installation will change things for you. It installs the Visual Basic Compatibility Layer (VBCL) for Access 2.0. So...

DO NOT INSTALL THIS PACKAGE IF YOU WANT TO RETAIN ACCESS 1.X WITH VISUAL BASIC!

On the other hand, installing this package will convert Visual Basic's data access features so they are compatible with Access 2.0 databases, including the new "Jet Engine" features of Access 2.0. This kit does not include the complete VBCL kit, however. It includes only those DLLs needed to access native Access 2.0 databases (.MDB), and not the "installable ISAM" files for BTrieve, FoxPro/XBase, or Paradox. To obtain a full copy of VBCL, download COMLYR.EXE from CompuServe (GO MSL), from the Microsoft Download Service at (206) 936-6735, or FTP from ftp.microsoft.com in /softlib/mslfiles. Note that the VBCL kit does *not* include the Access 2.0 (Jet 2.0) DLL. It is included with VBStats because I have a full Access 2.0 and ADK license, and therefore I am permitted to distribute it to "end users" as part of this application.

Getting Started:

BEFORE DOING ANYTHING ELSE, run the Setup Wizard (if you didn't choose to do this during installation). This establishes your entire environment.

If you didn't choose to have LogToDB automatically cycle your Windows httpd logfiles, make sure there's a server log file at the location you specified in the Setup Wizard. You can use the LOGCYCLE.EXE program that comes with Windows httpd to manually cycle the logs without stopping the server. If you're reporting on a CERN or NCSA server's log, move it into the approproate location.

Now run LogToDB. This will create the database (if needed) and fill in the data from the server log. This will probably take a fairly long time. See the description below for the reasons.

Next, run the Restriction Editor. Think carefully about what you want restricted and how. A common practice is to restrict *.GIF from the top-10 lists.

Finally, run the Reporter. Don't get impatient. This program can run for a long time, especially on a large database (one with tens of thousands of records). The restrictions are applied at report generation time, *not* at data gathering time. This feature requires complex queries. See the info below for a more complete explanation.

Look at the generated reports and the graph.

The remainder of this document describes system details. It is probably interesting only to those of you who want to customize the reports and/or add new reports and graphs.

The Database

The database is the heart of this package. It contains both tables and QueryDefs, the latter are stored SQL used to produce the summary report data, and perform maintenance tasks. Here's a brief description of them.

Table	Description
Accesses	Each access to the server, from a line in the log
AuthUsers	Each unique user that authenticated via the server
Objects	Each unique URL target that was accessed
PastTotals	Weekly totals generated by previous report runs
RestrictPats	Contains patterns used for visiblity restriction
RestrictKey	(not used, see below)
Sites	Each unique client hostname or IP address
TableKey	(not used, see below)

The RestrictKey table was supposed to contain a textual description of the restriction codes (more on this later). The TableKey table was supposed to contain a textual description of each table, listed by "table

code" used to identify tables during restriction processing. (Baffled? Don't worry... this is fun!)

There is always a User with ID=1, called <none>, which links with unauthenticated accesses.

Most of the SQL queries are in the database as QueryDefs. Many take parameters ("parameter queries"), most of which are the start and end date/times for the query. Doing this allows changes to be made to the database structure (and the query) without changing the code in the applications. Cool, eh? Well, there are a few SQL statements in the code anyway, so it ain't pure. Maybe I'll move them into querydefs some day. Here are brief descriptions of the QueryDefs:

QueryDef	Resulting dynaset (D) or action (A)
AuthUsersByAccessCount	(D) Total accesses for each user
AuthUsersByByteCount	(D) Total byte count for each user
MarkNewAuthUsers	(A) Init restrict code for newly added users
MarkNewObjects	(A) Init restrict code for newly added targets
MarkNewSites	(A) Init restrict code for newly added sites
ObjectsByAccessCount	(D Total accesses for each URL target
PatListSelector	(D) Restrict pattern lists for each table
PurgeAccesses	(A) Remove logged accesses for a date range
RemoveAllRestrictions	(A) Unrestrict all users, targets and sites
RestrictAuthUsers	(A) Apply restriction patterns to users
RestrictNewAuthUsers	(A) Apply restrictions to newly added users
RestrictNewObjects	(A) Apply restrictions to newly added targets
RestrictNewSites	(A) Apply restrictions to newly added sites
RestrictObjects	(A) Apply restriction patterns to targets
RestrictSites	(A) Apply restriction patterns to sites
SitesByAccessCount	(D) Total accesses for each site hostname/IP
SitesByByteCount	(D) Total byte count for each site hostname/IP
TotalAccessesByMethod	(D) Total accesses for each method (GET, etc.)
TotalAccesses	(D) Grand total of accesses
TotalByteCount	(D) Grand total of byte count
TotalIndexQueries	(D) Grand total ISINDEX requests

Some QueryDefs may be obsolete. They will be removed later. Here is the ObjectsByAccessCount SQL:

```
PARAMETERS pStart DateTime, pEnd DateTime;
SELECT DISTINCTROW Count(*) AS Accesses, Objects.ObjectName AS Object
FROM ((Accesses
    INNER JOIN Objects ON Accesses.ObjectID = Objects.ObjectID)
    INNER JOIN AuthUsers ON Accesses.AuthUserID = AuthUsers.AuthUserID)
    INNER JOIN Sites ON Accesses.SiteID = Sites.SiteID
WHERE ((Accesses.ObjectID = [Objects].[ObjectID])
     AND (Accesses.Time >= [pStart] And Accesses.Time < [pEnd])
     AND (Objects.Restrict = 0)
     AND (Sites.Restrict < 2)
     AND (AuthUsers.Restrict < 2))
GROUP BY Objects.ObjectName
ORDER BY Count(*) DESC;
```

The "ID"s are counter-generated links between the tables. The complexity arises out of the restriction capability.

First note the PARAMETERS are start and end time. The code sets these to control the time interval over which the totals are accumulated.

The SELECT part returns the access total for each "object" (URL target), GROUPed BY the target name

and ORDERed BY the access count, DESCending. So the dynaset is a list of access counts for each URL target, sorted by access count, in descending order. The first 10 rows of this are the "Top-10" accessed URLs and the number of accesses for each.

The first INNER JOIN links to the Objects table so we can get the URL target name instead of the internal ID which is in the Accesses table. It also lets us get the restriction code for the object. The second and third INNER JOINs are used to access the AuthUser and Site tables in order to get the restriction codes for the site and the user involved in the access records that make up the total for each of the objects.

Here's the trick on restrictions. If the object's restriction code is 0, it always counts. If it is 1, the object is hidden from top-10 lists. If it is 2, the object is ignored altogether.

The implications of this policy are subtle. For example, given an access to /foo/bar.html by user farkle from site dorfle.com, this access will be counted in the totals only if all three objects (URL target, user, and site) have restriction codes of less than 2. If either user farkle or site dorfle.com have restriction codes of 2, the access will be ignored. In addition, if the target /foo/bar.html has a restriction code of greater than 0, it will not be counted in the top-10 URL LIST. However, that access will be counted in the top-10 SITE LIST and top-10 USER LIST unless either farkle or dorfle.com are restricted greater than 0.

Look at the WHERE clause (after the linking phrase): First, the date range is applied. Next the top-10 restriction is applied to the primary object (URL target), then the total-hiding restriction is applied to the secondary objects (user and site).

The Restriction List Editor handles the application and removal of restrictions so that the user needn't be concerned with all of this gory detail. Thank God. Meanwhile, you will get probably confused by all of this in the beginning.

Note that ALL accesses are kept in the database. Restrictions may be changed at any time. Subsequent reports will reflect the new restrictions. Consider the alternative where access records are filtered at the time they are brought into the database.. What if you change your mind later??? Maybe you can see why SQL is so cool. Consider doing this in C or perl! (I can already hear the snickers from dedicated perl hackers out there).

The PastTotals table is used to prevent re-doing reports already done, so if you change restrictions, use the Maintenance Utility to purge the PastTotals table before re-running Reporter.

I'm out of breath, so let's press on to the description of the "worker" programs:

LogToDB

This program handles the importing of Web server Common Log Format files into the Access database. It will create the empty database if it doean't exist.

Unless you disabled it, LogToDB will perform DNS reverse lookups on the IP addresses in the flat file log, converting them to hostnames (where possible). This process is paced at a maximum of 3 lookups per second, to be friendly to the net. So be patient, LogToDB may run for a long time if you have thousands of entries in your log.

If you specified "cycle logs" in the Setup Wizard, it expects to be run on the same machine as a Windows httpd web server, with the server running. It first signals the server to cycle its logs using the SIGHTTPD.DLL. Then it waits for the cycled-out log to appear in the server's logs directory as specified in the Setup Wizard.

In any case, it opens the text logfile and imports the data into the database, which it will create if needed with the path and name you specified in the Setup Wizard..

LogToDB may be run at any time, as often as desired. It will simply pull the data out of the server's textual log into the database (normalizing the data structure). It is designed so it can be run from a scheduler so it needs no human interaction.

If LogToDB encounters a corrupted log record, the record is skipped. An icon is displayed while the program runs, indicating the number of records processed, in multiples of 10. No effort is made to detect duplicate access records, however unique sites (hostname or IP address), URL targets and authenticated users are kept in separate tables, and the access records link to these other tables via an internally generated 32-bit ID.

Source Note: The BUILD_DB module was generated from a working copy of the database that I originally created with Access. I used a program called DB2BAS, which I hacked to generate BASIC that creates the QueryDefs as well as the tables and fields. The hacked DB2BAS did much of the job, but I still had to edit the result by hand. Any changes you make to the database will have to be reflected in BUILD_DB by hand. Sorry. The hacked DB2BAS is a mess. I'm not sure it really saved me all that much time.

Rescriction Editor

This is a GUI program that is used to maintain the restriction specifications for the database. There is online help available. The match patterns used in the restrictions are those of the Access/VB SQL language, and are documented in the help. A small help button next to the pattern entry field pops up a pattern reference window. The patterns are kept in the RestrictPatterns table in the database.

Once you have made the changes to restrictions, you must choose Apply Restrictions in the Database menu. This removes old restrictions and uses the patterns to set the restriction codes appropriate to the changed patterns and/or restriction levels. You can also remove restrictions completely by choosing Remove Restrictions in the database menu.

Reporter

This program generates the statistics HTML pages, and a Windows BMP graph of total accesses by week. The report HTML documents, and the BMP graph are put into the location you specified in the Setup Wizard. Needless to say, this location must be accessible from URL space if you want your Web server to provide the reports to clients!

The program may be run at any time. It will produce reports for all complete weeks from the first day for which information appears in the database through the just-completed week. Nothing occurring after last Sunday is reported because the week isn't over yet. In order to prevent duplication of effort, the reporter records summary infrmation for each report it generates in the PastTotals table. The record number is used as the "tag" on the HTML filenames. Next time the reporter runs, it looks at PastTotals and skips data for which reports have already been generated.

NOTE: If you want to regenerate reports (perhaps you changed the restrictions and want past reports re-run), delete all of the PastTotals records and re-run the reporter. You can use Visual Data to do this. All past reports will be regenerated. Note that the report numbers will not start back at 1 unless you compress the database after deleting the PastTotals data.

The reporting process is straightforward. It uses the QueryDefs to produce the statistics information, then takes that and formats it into HTML reports, one per week. It produces daily GIF access graphs for each week using the VB Graph control and the BMP2GIF DLL, and adds a link to it to the weekly report.

At the end, it produces an "index" page, using the data in the PastTotals table to produce a graph showing the total accesses by week, again using the VB Graph control. The image is automatically written to disk in

Windows DIB format (".BMP" file). Then it calls BMP2GIF.DLL to convert the BMP-format graph into GIF format. The previously generated index page contains a HREF link to the GIF format graph.

The program may be run from a scheduler, as it requires no human interaction. As it runs, an icon displays the week-starting date of the report being produced.

Maintenance Utility

This program automates three mundane tasks. With it you can purge the database of "old" access records. Unless you do this, the database will grow indefinitely. It also can clear the PastTotals table. This has the effect of forcing the Reporter to regenerate all reports for which there are data in the database. This is useful when you change restrictions, and want to re-run reports based on the new restriction criteria. Lastly, it can "compact" the database. You probably want to do this after doing a purge of old Access data. If you are really picky, you'll notice that, after clearing PastTotals, the regenerated reports' file names aren't numbered starting with 1. You must compact the database to reset the internal counters used as the source of those sequence numbers.

Ideas for the Future

You gan get the sources, DO IT!

Add some more graphs. It's easy! Thumbnails, other statistics...